VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

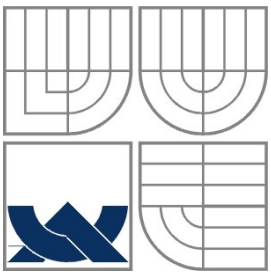
BENCHMARKING IMPLEMENTACÍ DNS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

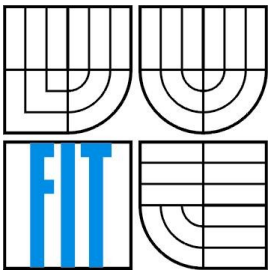
AUTOR PRÁCE
AUTHOR

MARTIN HALLER

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

BENCHMARKING IMPLEMENTACÍ DNS

BENCHMARKING OF DNS IMPLEMENTATIONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MARTIN HALLER

VEDOUCÍ PRÁCE
SUPERVISOR

ING. MATOUŠEK PETR, PH.D.

BRNO 2010

Abstrakt

Bakalářská práce vznikla ve spolupráci se sdružením CZ.NIC (správcem TLD domény CZ). Jejím cílem bylo prozkoumat existující metodiky benchmarkingu autoritativních implementací DNS, rozšířit je a pomocí nich protestovat vybrané implementace DNS.
Bakalářská práce obsahuje souhrn současně dostupných metodik testování autoritativních DNS serverů. Tyto metodiky byly zanalyzovány, přepracovány a doplněny, ve výsledku vznikla nová komplexní metodika.
Byly vybrány známé a populární open source implementace autoritatívních DNS serverů BIND, NSD, PowerDNS a MaraDNS. Vybrané implementace byly protestovány navrženou metodikou a výsledky vyhodnoceny.
Metodika i výsledky byly prezentovány na mezinárodním semináři sdružení DNS-OARC, které sdružuje ty nejlepší experty a organizace z oblasti DNS, v Praze, kde vzbudily pozitivní reakce.

Abstract

This bachelor thesis has been made with assistance of CZ.NIC (Czech TLD administrator) association. Goal of this thesis was to examine methodology of authoritative DNS servers benchmarking, extend it and test it.
The bachelor thesis contains overview of current DNS benchmarking methodologies. The current methodologies were analyzed, revised and extended. As a result a new complex methodology was proposed.
Popular open source DNS implementations were selected and benchmarked by the new methodology. Results of the benchmark were discussed.
The new methodology and the results were presented at DNS-OARC workshop. The methodology and the results were rated very well.

Klíčová slova

Výkonové testování, DNS, BIND, NSD, PowerDNS, MaraDNS

Keywords

Benchmark, DNS, BIND, NSD, PowerDNS, MaraDNS

Citace

Martin Haller: Benchmarking implementací DNS, bakalářská práce, Brno, FIT VUT v Brně, 2010

Benchmarking implementací DNS

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ondřeje Surého ze sdružení CZ.NIC a Ing. Petra Matouška, Ph.D.

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Martin Haller
18.5.2010

Poděkování

Rád bych poděkoval svému vedoucímu práce panu doktoru Petru Matouškovi za vlídné a přátelské vedení mé bakalářské práce. Také bych rád poděkoval panu Ondřeji Surému ze sdružení CZ.NIC za poskytnutí odborných rad a příležitosti prezentovat mou práci na mezinárodní konferenci, kde jsem získal zpětnou vazbu a nové nápady.

Obsah

1 Úvod

DNS je nedílnou součástí dnešních sítí včetně internetu. Zejména internet tak, jak ho v současné době zná většina lidí, je na této službě funkčně závislý.

Jeho potřebnost stále stoupá stejně jako jeho zatížení. Vždyť i pro jedno načtení webové stránky se běžně generují desítky DNS dotazů. Naštěstí se tvůrcům podařilo navrhnout DNS systém velmi snadno rozšiřitelný do dvou směrů. Prvním směrem je „vysoká dostupnost“ (high avalaibility) a druhým je „vysoká výkonnost“ (load balancing).

Díky zátěži, jaká je na systém DNS kladena, je každý správce DNS serveru rád za optimalizaci systému DNS či zvýšení jeho výkonnosti. Z tohoto důvodu vzniklo téma této bakalářské práce.

Má bakalářská práce vznikla ve spoluprací se zájmovým sdružením právnických osob CZ.NIC (www.nic.cz), které je správcem TLD (top level domain) domén CZ a části zóny ENUM. CZ.NIC se zajímá o výkonnostní srovnání různých implementací DNS, které by mu mohlo pomoci zvýšit efektivitu a kvalitu služeb do budoucna.

Cílem této bakalářské práce je zanalyzovat a rozšířit současné metodiky výkonového testování implementací DNS. S pomocí nalezených metodik provést testování vybraných implementací DNS a výsledky vyhodnotit.

Bakalářská práce začíná kapitolou 2 Současný stav, ve které analyzuji současný stav v oblasti výkonového testování autoritativních implementací DNS serverů. Kapitola obsahuje rozbor existujících metodik a jejich výsledků.

První ze dvou stěžejních kapitol je zejména teoretická kapitola 3 Metodika testů DNS serverů. V této kapitole rozebírám již existující metodiky a na jejich základech se snažím vytvořit novou metodiku, která je lepší a komplexnější. Při návrhu nové metodiky procházím fázemi návrhu jako jsou výběr a definování požadovaných vlastností, měřitelných ukazatelů, postupů, pravidel pro měření a podoby prezentace výsledků. Kapitola je zakončena zhodnocením navržené metodiky.

Druhou stěžejní kapitolou, nyní však již praktickou, je kapitola 4 Praktické testování. V této kapitole popisuji jak prakticky implementovat (provádět) metodiku definovanou v předchozí kapitole. Popisuji vytvořené nástroje a modelovou konfiguraci. Vytvářím souhrn populárnosti autoritativních implementací DNS serverů a dle něho vybírám implementace k testu. Kapitola obsahuje výsledky a grafy naměřené s pomocí definované metodiky. Všechny výsledky jsou vyhodnoceny, rozebrány a okomentovány. V závěru kapitoly souhrnně vyhodnocuji testované implementace

Bakalářská práce je zakončena závěrem obsahujícím zkušenosti a postřehy získané během implementace metodiky a praktického testování. V závěru jsou také uvedeny možnosti dalšího rozvoje této bakalářské práce.

Výsledky bakalářské práce jsou určeny všem správcům autoritativních DNS serverů, kterým ušetří čas a peníze při rozhodování, kterou implementaci zvolit, jelikož v současné době není takto komplexní test autoritativních implementací DNS na internetu k nalezení.

Dne 1.5.2010 mi bylo umožněno prezentovat výsledky mé bakalářské práce na mezinárodním semináři výzkumného sdružení DNS-OARC[1] v Praze. Na tomto semináři jsem získal zpětnou vazbu, nové nápady a postřehy, které jsem se pokusil zpětně zapracovat do této práce.

1 Sdružení DNS-OARC sdružuje nejvýznamnější organizace a osoby z oblasti DNS. Jeho smyslem je hlavně výzkum, vývoj a monitorování DNS. Více na domovské stránce https://www.dns-oarc.net/.

2 Současný stav

Při prohledávání internetu a elektronických technických knihoven se mi nepodařilo narazit na něco opravdu použitelného, od čeho bych se mohl odrazit. Problém vidím v tom, že dané téma výkonnostního testování (benchmarkingu) implementací autoritativních DNS serverů je zajímavé pouze pro úzký okruh lidí. Mezi tento okruh lidí se řadí pouze správci TLD domén a ti největší poskytovatelé autoritativních DNS služeb.

Na internetu je možné najít mnoho benchmarků a nástrojů zaměřených na zájmy obyčejných internetových uživatelů, jakožto vyhodnocování rychlosti odpovědí jednotlivých DNS serverů. To je zajímavé pro uživatele, protože jim to může pomoci k rychlejšímu načítání webových stránek. pro tuto bakalářskou práci to zajímavé není, a to proto, že zpoždění je způsobené zejména zátěží serveru a zpožděním linky. Ve výše zmíněných nástrojích či benchmarcích se ani nepočítá se zjišťováním implementace DNS na měřeném serveru.

2.1 Benchmark „The choices for a nameserver“

Nejvíce relevantním benchmarkem, který se mi podařilo najít, je „The choices for a nameserver“[1] od Stephane Bortzmeyer. Jeho test má stejné zaměření (pouze implementace autoritativního DNS), je jednoduchý a přehledný. Obsahem benchmarku jsou implementace NSD, BIND a PowerDNS (jež jsou také součástí benchmarku této bakalářské práce). Stephane Bortzmeyer testoval tyto implementace na dvou rozdílně velkých zónách a dospěl k výsledkům shrnutým v tabulce 2.1.

velikost zóny (domén)	150 000	870 000
implementace	**počet odpovědí za sekundu**	
BIND	9 305	6 948
NSD	27 822	33 155
PowerDNS	197	netestován

Tabulka 2.1: Výsledky výkonnostního testu od Stephane Bortzmeyer

Naprostým vítězem tohoto benchmarku se stala implementace NSD. Bohužel autor neuvedl verze jednotlivých testovaných implementací a benchmark je již v době psaní této práce přes šest let starý, takže je velmi možné, že moje výsledky budou naprosto rozdílné.

Benchmark nebyl dokonalý a rozpoutal na internetu i drobné diskuze. Nejvíce byla pobouřena komunita okolo implementace PowerDNS, která v testu propadla. Autorovi bylo vyčítáno, že neprovedl žádné výkonnostní ladění této implementace, která není ve výchozím nastavení pro takovéto testy připravena. Díky tomu PowerDNS nemohl konkurovat a během testu u něho docházelo i ke ztrátě dotazů (resp. nestíhal je zodpovídat). Po těchto diskuzích a stížnostech i sám autor benchmarku dodal k výsledkům poznámku ohledně PowerDNS.

2.2 Benchmark „DNS Name Server Comparison“

Dalším poměrně relevantním testem je „DNS Name Server Comparison“[2] od autora jménem Brad Knowles. Bohužel tento benchmark je o něco starší a chaotičtější. Autor v tomto benchmarku dělá

průzkum podílu jednotlivých implementací DNS na internetu. Následně měří rychlost a četnost různých nastavení rekurzivních serverů. V závěru se zaměřuje i na autoritativní DNS servery, ale prezentace zde obsahuje pouze nepopsané grafy, ze kterých se mi nepodařilo nic vyčíst.

3 Metodika testů DNS serverů

Metodika je „nudnou“ teoretickou prací, kde se specifikují testy, postupy a výstupy. Bohužel bez metodiky by samotný benchmark neměl žádný smysl a přínos. V této kapitole tedy navrhnu metodiku pro provádění benchmarku.

Zjednodušeně návrh metodiky probíhá následovně:

1. Shromáždit již existující metodiky
2. Zanalyzovat již existující metodiky
3. Zanalyzovat, co všechno může být zajímavé měřit, testovat či znát
4. Dát dohromady to nejlepší ze všech metodik (co nám přijde užitečné a v souladu s bodem 3.)
5. Rozšířit výslednou metodiku o to, co chybělo v předchozích metodikách, tak abychom naplnili bod 3.

Tímto postupem se dá vytvořit slušná metodika. Předpokladem je, že tvůrce je expertem s hlubokými znalostmi dané problematiky. Samozřejmě dalším zádrhelem bude tuto metodiku prakticky implementovat.

3.1 Existující metodiky

Pro analýzu existujících metodik se odkážu na kapitolu 2 Současný stav, v té kapitole jsem již vyhledal relevantní existující metodiky a testy, které tu rozeberu.

Nejprve rozeberu metodiku testu „The Choices For a Namerserver“[1]. Ukazatelem v této práci byl pouze maximální počet dotazů za sekundu, který je implementace schopná zodpovědět. Test proběhl ve dvou variantách, jednou implementace obsluhovaly zónu o 150000 delegovaných doménách a podruhé o 870000 delegovaných doménách. S pomocí programu Queryperf se pak zaslal každé implementaci jeden milion dotazů[2] a měřila se doba, za kterou je zvládly zodpovědět (z toho se vyvodil počet dotazů za sekundu). Před každým testem byly implementace čerstvě spuštěny a počítače byly propojeny pomocí kroucené dvoulinky skrze síťový přepínač.

Z druhého testu není metodika ani výsledky v oblasti autoritativních DNS serverech patrná, a proto je zde nemohu rozebrat.

3.2 Návrh nové metodiky

Bohužel kvůli nedostatku existujících metodik (mnou nalezených a mně známých) jsem neměl od čeho se odrazit a musel si vše vytvořit téměř od začátku.

2 Složení dotazů bylo následující: 75% dotazy na existující doménová jména, 25% dotazů na neexistující doménová jména.

3.2.1 Požadované vlastnosti

Nejprve je důležité si ujasnit, jak má metodika vypadat. První vlastností benchmarku musí být *simulace reálných situací*, aby výsledky benchmarků byly užitečné. Poslední dobou se zdá, že se provádí spousta benchmarků testujících situace, které se v reálném světě nevyskytují, a benchmarky jsou upraveny pro potřeby určitých zájmových skupin (např. společností) tak, aby v nich jejich favoriti vyhráli. Kvůli takovým situacím vznikla obecná tvrzení ve stylu „existují dva druhy lží, lži a benchmarky".

Další vlastností je *komplexnost***,** je důležité testovat všechny zajímavé situace a parametry. Bez komplexních testů by se mohlo stát, že by v benchmarku vyhrála implementace XY, která by ovšem byla prakticky nepoužitelná, protože by měla slabiny v něčem, co se neměřilo, ale co je důležité pro skutečný provoz.

Důležitá je také *porovnatelnost* výsledků benchmarku mezi sebou. To znamená, že je třeba zabezpečit, aby všechny implementace měly stejné podmínky testu a test byl co nejvíce identický. Za splnění těchto podmínek jsou mezi sebou výsledky benchmarku více porovnatelné a vypovídající.

Posledním parametrem je *behaviorálnost* (monitorování chování). Tato vlastnost by se určitě dala schovat i pod komplexnost, ale pro její důležitost ji uvádím zvlášť. Je třeba, aby část benchmarku hodnotila i chování implementací. Myšleno je to tak, aby se kromě měření „statických" hodnot (maximální, minimální, průměrné) měřila i například závislost (charakteristika) využití CPU na počtu dotazů za sekundu.

3.2.2 Co měřit

Na základě již publikovaných benchmarků a mé analýzy problematiky jsem si vypracoval následující tabulku 3.1 obsahující možné varianty benchmarku, proměnné a ukazatele.

Varianty	**Proměnné**	**Ukazatele**
UDP	Dotazů za sekundu	CPU
DNSSEC	Zónové soubory	Operační paměť
NSEC3	Délka klíče (NSEC3)	Disk
TCP	NSEC3 iterací při tvorbě hashe (NSEC3)	Zpoždění
		Procesy / vlákna
		Ztracené dotazy
		Velikost odchozího datového toku
„Statické" testy		
Doba startu		
Doba načítání změn		
Maximální počet dotazů za sekundu		

Tabulka 3.1: Možné varianty testů, měřitelné hodnoty (ukazatele) a proměnné testů

Tabulka obsahuje možné varianty benchmarku. pro každou tuto variantu by se měl dělat benchmark zvlášť. Z možných variant je patrné, že výčet není plně specifikován, jelikož se musí nakombinovat UDP či TCP s variantou DNSSEC, NSEC3 nebo bez. Toto je uděláno schválně pro větší přehlednost, protože není třeba testovat zároveň UDP DNSSEC a TCP DNSSEC a protože rozdíl by u nich měl být stejný jako mezi UDP a TCP bez DNSSECu.

Proměnné představují to, co se při benchmarku nastavuje. Jedná se tedy o dvě základní proměnné a rozšiřující dvě proměnné, které jsou k dispozici pouze při použití NSEC3.

Posledním sloupcem v tabulce jsou ukazatele, které reprezentují to, co nás zajímá a co máme měřit. Předpokládám, že důvod pro měření všech ukazatelů je zřejmý, a proto ho zde nebudu rozebírat.

3.2.3 Způsob měření

Při měření by měla být dodržována určitá pravidla. Níže je uveden výčet těch nejdůležitějších, na které jsem narazil. Rád bych podotkl, že výčet není rozhodně konečný a nová pravidla mohou být objevena během praktikování těchto testů. Není totiž možné dát dohromady „vše" bez zkušeností.

- Před každým testem je potřeba znovu spustit implementaci či celý systém (provést restart), pokud situace nevyžaduje jinak.
- V případě, že se při testování objeví více jak jeden bottleneck, je třeba testy upravit, aby došlo ke snížení počtu bottlenecků pouze na jeden. V případě, že testující ví, co dělá, a více bottlenecků je záměrem a nebo nebrání interpretaci výsledků, pak může být toto pravidlo ignorováno.
- Minimální počet fyzických počítačů pro provedení benchmarku je dva.
- Implementace DNS musí být provozována na jiném PC než na zatěžujícím počítači.
- Propojení počítačů musí být spolehlivé, aby nedocházelo ke ztrátě provozu, a stabilní, aby na něm nedocházelo ke kolísání rychlosti a latence (zpoždění).
- Na počítači provozujícím implementaci DNS musí být vypnuty všechny služby nepotřebné pro běh benchmarku.

3.2.4 Podoba testů

V této kapitole je tedy specifikována podoba benchmarku, která vychází z předchozích kapitol. Benchmark je rozdělen na „statické" a behaviorální testy. Všechny tyto testy by měly být provedeny pro každou modelovou variantu specifikovanou tabulkou 3.1.

Provedení všech „statických" testů uvedených v tabulce 3.1. Prezentace výsledků je pouze numerická (např. zaznamenaná v tabulce). V případě zájmu mohou být tyto testy provedeny s proměnnou velikostí zón a výsledky zobrazeny v grafu jako závislost na velikosti zón.

Pro každý ukazatel by mělo být provedeno co největší množství měření s různými hodnotami proměnných a výsledek zobrazen ve trojrozměrném grafu. V případě testování implementací s funkčním rozšířením NSEC3[3] vytvořit dvojrozměrné pole trojrozměrných grafů, či případné jiné vhodné grafické znázornění.

3.3 Závěr

V této kapitole jsem specifikoval metodiku. Výsledná metodika je výsledkem mého přesvědčení, mých znalostí, mé upřímné snahy a konzultací s panem Ondřejem Surým (odborníkem z CZ.NIC).

3 Rozšíření DNSSECu, nahrazující záznamy typu NSEC za nové záznamy typu NSEC3, které již nesdělují doménová jména ve formě čistého textu. Více například v RFC 5155 (http://www.ietf.org/rfc/rfc5155.txt).

4 Praktické testování

V celé kapitole 3 Metodika testů DNS serverů jsem se snažil rozebírat současné metodiky a navrhnout novou. V této kapitole popisuji, co všechno se mi podařilo z mnou navrhované metodiky implementovat (změřit) a jak. Dále zde popisuji věci, které se mi bohužel nepodařilo implementovat (změřit) a proč. Kapitola dále obsahuje výsledky se slovním zhodnocením.

4.1 Podoba praktických testů

Z mnou navrhované metodiky jsem se toho snažil implementovat co nejvíce, avšak někde jsem narazil na technologické problémy, neexistující nástroje a enormní časovou náročnost.

Varianty	**Proměnné**	**Ukazatele**
UDP	*Dotazů za sekundu*	*CPU*
DNSSEC	Zónové soubory	*Operační paměť*
NSEC3	Délka klíče (NSEC3)	*Disk I/O*
TCP	NSEC3 iterací při tvorbě hashe (NSEC3)	*Zpoždění*
		Procesy / vlákna
		Ztracené dotazy
		Velikost odchozího datového toku
„Statické“ testy		
Doba startu		
Doba načítání změn		
Maximální počet dotazů za sekundu		

Tabulka 4.1: Zobrazení implementovaných částí metodiky

Tabulka 4.1 je kopií tabulky 3.1, která navíc obsahuje zobrazení úspěšně implementovaných věcí (zeleně a kurzívou). O úspěšně implementovaných věcech se rozepisuji dále u výsledků, a tak v této kapitole shrnu pouze ty neimplementované a důvody, proč jsem je neimplementoval.

Začnu tedy u neimplementovaných variant. *NSEC3* nebyl implementován z časových důvodů, jelikož jeho implementace by několikanásobně zvýšila časovou náročnost testů. V případě začlenění NSEC3 by bylo nejlepší vytvořit automatizované nástroje, které by samy dokázaly vytvářet, nastavovat, spouštět a vyhodnocovat testy. Takovéto řešení by ovšem dle mne bylo složitostí na úrovni diplomové práce.

TCP varianta nebyla implementována z důvodů problémů s nedostatečným počtem TCP soketů[4] a neexistujících nástrojů[5].

Co se týče neimplementovaných proměnných (*zónové soubory*, *délka klíče*, *NSEC3 iterace*), tak tam platí to samé jako u NSEC3.

Ztracené dotazy nejsou implementovány z důvodu nedostatečných nástrojů. Současné nástroje, které jsem používal a upravoval, nebyly schopny podávat adekvátní výstup, abych mohl tento ukazatel měřit. Do budoucna bych viděl měřitelnost tohoto parametru pomocí nového měřícího programu, který by bylo třeba vytvořit.

4 Problémem je, že linuxové jádro neumožňuje vytvoření neomezeného počtu TCP soketů a právě uzavřený soket může být znovu využit až po určité době viz RFC973 (http://tools.ietf.org/html/rfc793#section-3.5).

5 I když se mi podařilo sestrojit UDP/TCP konvertor, abych mohl využít stávající UDP nástroje.

Měřit *velikost odchozího datového toku*, mi bylo doporučeno až na konferenci OARC, kde jsem prezentoval výsledky. Datový tok se může lišit z důvodu používání volitelné komprese zpráv DNS. Rozdíl by mohl být výrazný zejména při používání dlouhých lokalizovaných doménových jmen (IDN). Konference se bohužel konala velmi blízko termínu odevzdání bakalářské práce a nebylo z časových důvodů možné tento parametr začlenit.

Dobu načítání změn jsem se nakonec rozhodl neměřit, jelikož předpokládám stejnou dobu jako u doby startu. Navíc není tato informace natolik důležitá, jelikož TLD si to řeší tak, že primární DNS server je privátní, na něm se změny v klidu promítnou a následně se pomocí IXFR[6] přesunou na sekundární veřejné DNS servery.

4.2 Nástroje

Pro testování výkonu jednotlivých implementací DNS jsou třeba různé programy a nástroje. Já osobně jsem zastáncem toho, že je vždy lepší použít již existující řešení, než vyvíjet nové. Samozřejmě pokud existující řešení nejsou dostatečná a nebo existují jiné důvody pro vývoj vlastního řešení, pak je potřeba vyvinout si své nástroje. Já jsem při hledání narazil na již existující nástroje a tam, kde to šlo, tam jsem je použil.

Můj benchmark je poměrně komplexní (dle mého názoru), proto jsem si musel některé programy upravit a vytvořit vlastní skripty pro co největší automatizaci jednotlivých testů, aby mohly být kdykoliv a kýmkoliv co nejsnáze opakovány či rozšířeny o další implementace.

4.2.1 Queryperf

Jedná se o nástroj pro zatěžování DNS serverů a měření jejich výkonosti. Program je součástí balíčku BIND jako podpůrný program (contrib) a je vydán pod licencí GNU/GPL.

Program zasílá DNS dotazy, které čte ze souboru, na DNS server a přijímá odpovědi. Program sleduje zpoždění přijatých odpovědí (za jakou dobu dorazily) a ztracené odpovědi (resp. nezodpovězené dotazy). Program ve výchozím nastavení (bez použití určitých parametrů) testuje maximální propustnost DNS serveru (kolik je schopný zpracovat dotazů) a tuto naměřenou hodnotu zobrazí.

Program ke svému chodu potřebuje textový soubor, kde jsou uvedeny dotazy, na které se má ptát. Formát textového souboru s dotazy je takový, že na každém řádku je jeden dotaz skládající se z typu záznamu a doménového jména, na které se dotazovat. Tento soubor s dotazy nám umožní spustit pro každou implementaci totožnou sadu dotazů, čímž činí výsledky tohoto testu více porovnatelné.

Přidávání nových funkcí nebo úprava již existujících je poměrně náročná a někdy i nemožná. Program je totiž psán strukturovaně s použitím velkého množství globálních proměnných. Přesto se mi podařilo do programu doprogramovat funkce, co jsem potřeboval, a opravit některé existující chyby.

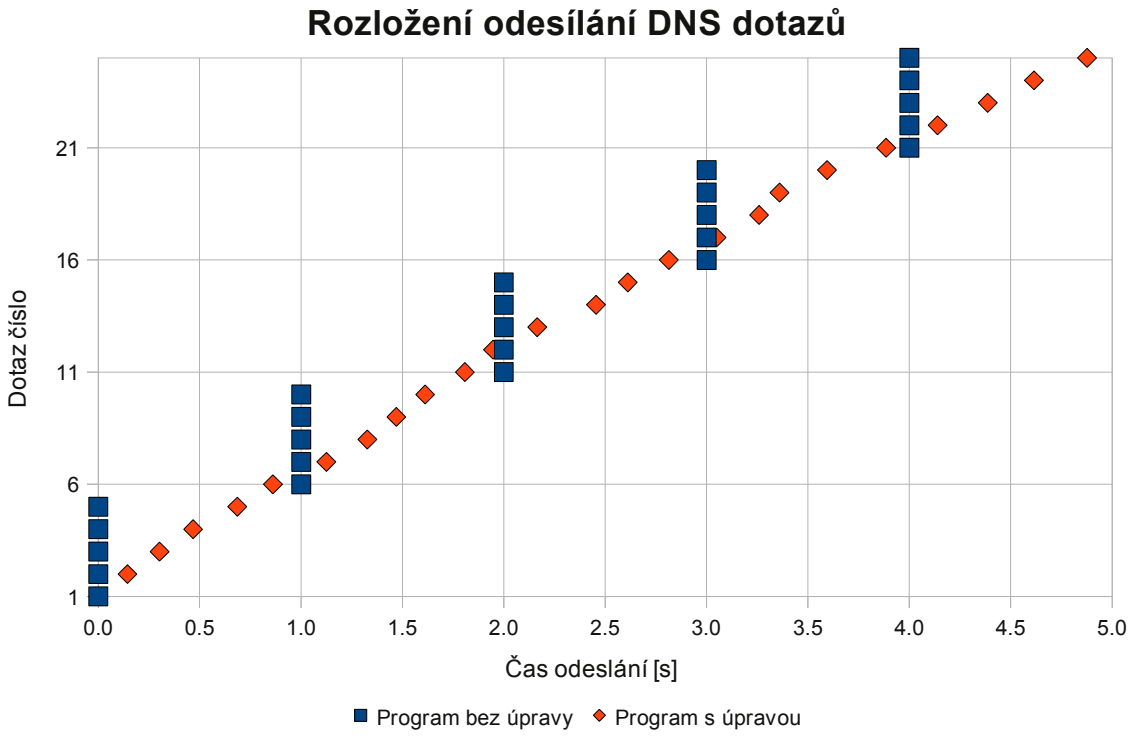
6 Incremental zone transfer viz http://www.dns.net/dnsrd/rfc/rfc1995.html.

4.2.1.1 Rozložení dotazů

S použitím určitých přepínačů umí program omezit počet dotazů, které za jednu sekundu vyšle. Což je to, co pro své behaviorální testování potřebuji. Ovšem při podrobnější analýze jsem u tohoto módu narazil na problém, že odesílá dotazy nárazově ve skupinách o velikosti dané parametrem „maximum number of outstanding queries“ (počet dotazů čekajících na odpověď). Takovéto chování se příliš nepodobá skutečnému provozu.

Díky tomu, že je program šířen jako „open source“, tak jsem mohl provést úpravu kódu a změnit chování programu na chování více podobné skutečnému provozu. Program si ze zadaného počtu dotazů za sekundu spočítá, kolik mikrosekund má být pauza mezi jednotlivými odesílanými dotazy. Díky tomu je zatížení serverů více podobné skutečnému provozu. Avšak pro ještě lepší simulaci reálného provozu se každé čekání upraví náhodným počtem mikrosekund, o které bude čekání delší nebo kratší.

Nejlépe se tato úprava pochopí z ilustrace 4.1. Tato ilustrace zobrazuje rozložení odesílání jednotlivých DNS dotazů programem bez úpravy a programem s úpravou. Programy byly nastaveny na běh po dobu pěti sekund s limitem 5 dotazů za sekundu. Data potřebná pro sestavení této ilustrace byla získána pomocí programu Wireshark[7].



Ilustrace 4.1: Charakteristika odesílání DNS dotazů programem v čase.

Bohužel při praktickém testování jsem tuto vlastnost musel odstranit. Tato úprava zvyšovala HW nároky na testující počítač a u jedné z DNS implementací nebylo možné dosáhnout jejího plného zatížení. I přesto, že již tato podkapitola není „platná“, nechám ji zachovanou jako nastínění možnosti, jak rozvíjet benchmark a možná úskalí.

7 Wireshark je program pro zachytávání a analýzu síťového provozu (http://www.wireshark.org/).

4.2.1.2 Změna parametrů za běhu programu

Samotný program umožňuje za svého běhu měnit určité parametry, a tím dynamicky měnit testování. Těmito parametry jsou adresa DNS serveru, port, počet nezodpovězených dotazů a časový limit pro zodpovězení dotazu. Jelikož chci testovat implementace DNS s postupným zvyšováním zátěže, potřeboval jsem ještě možnost měnit parametr „počet dotazů za sekundu“.

Nezbylo mi nic jiného než tuto vlastnost doprogramovat. To se mi nakonec podařilo a budu se snažit tuto změnu reportovat a začlenit ji do oficiálního kódu nástroje Queryperf.

4.2.2 Monitorovací software

Původně jsem chtěl použít monitorovací software Munin[8], se kterým mám již zkušenosti. Munin díky řadě modulů dokáže sledovat spoustu systémových proměnných a prostředků. Naměřené hodnoty si zaznamenává a pravidelně vykresluje přehledné grafy. Nakonec jsem se rozhodl ho nepoužít z důvodu nemožnosti změny frekvence měření a nemožnosti generovat grafy za určitou dobu.

Z výše uvedeného důvodu jsem se nakonec rozhodl napsat si vlastní program pro monitorování systému. Abych nepsal vše od začátku, vypůjčil jsem si moduly z programu Munin a napsal si vlastní monitorovací skript. Skript je napsaný v prostředí BASH s využitím standardních linuxových programů (nástrojů). Skript běží po předem zadanou dobu, během které monitoruje stav systému, a po svém skončení vygeneruje přehledné grafy.

Skript si po svém startu vytvoří databáze RRD (round robin database) pomocí programu RRDtool[9] a po předem stanovenou dobu provádí cyklické monitorování systémů.

Jak jsem se již zmínil, systémové ukazatele (stav operační paměti, vytížení CPU atd.) jsou získávány pomocí modulů vypůjčených z programu Munin. Tyto moduly většinou pracují tak, že čtou speciální „soubory“ vytvářené linuxovým jádrem a formátují je do podoby jednodušší pro další zpracování. Skript následně uloží jejich výstup do databáze RRD a sám sebe uspí do doby dalšího měření.

Jakmile uplyne doba, po kterou má program monitorovat systém, tak vytvoří skript pro tvorbu grafů, který následně spustí a ukončí se. Skript pro tvorbu grafů je vytvářen záměrně, a to pro jednoduší úpravu vzhledu grafů v budoucnu.

4.2.3 Nástroj TimeDnsRefresh.pl

Jedná se o nástroj z balíku „DLS Performance Tools“ projektu BIND DLZ[10]. Nástroj slouží k měření doby, za jakou se projeví změna v záznamech DNS. Dá se s ním tedy měřit doba startu DNS serveru či doba, za jakou DNS server zaktualizuje záznamy.

Nástroj funguje na bázi zasílání DNS dotazu zadaného jako argument a ověřování odpovědi oproti zadanému regulárnímu výrazu. Výstupem tohoto programu je pak doba, jaká uběhla od jeho spuštění k nalezení odpovědi vyhovující zadanému regulárnímu výrazu.

---

8 Munin je open source monitorovací systém (http://munin-monitoring.org/).

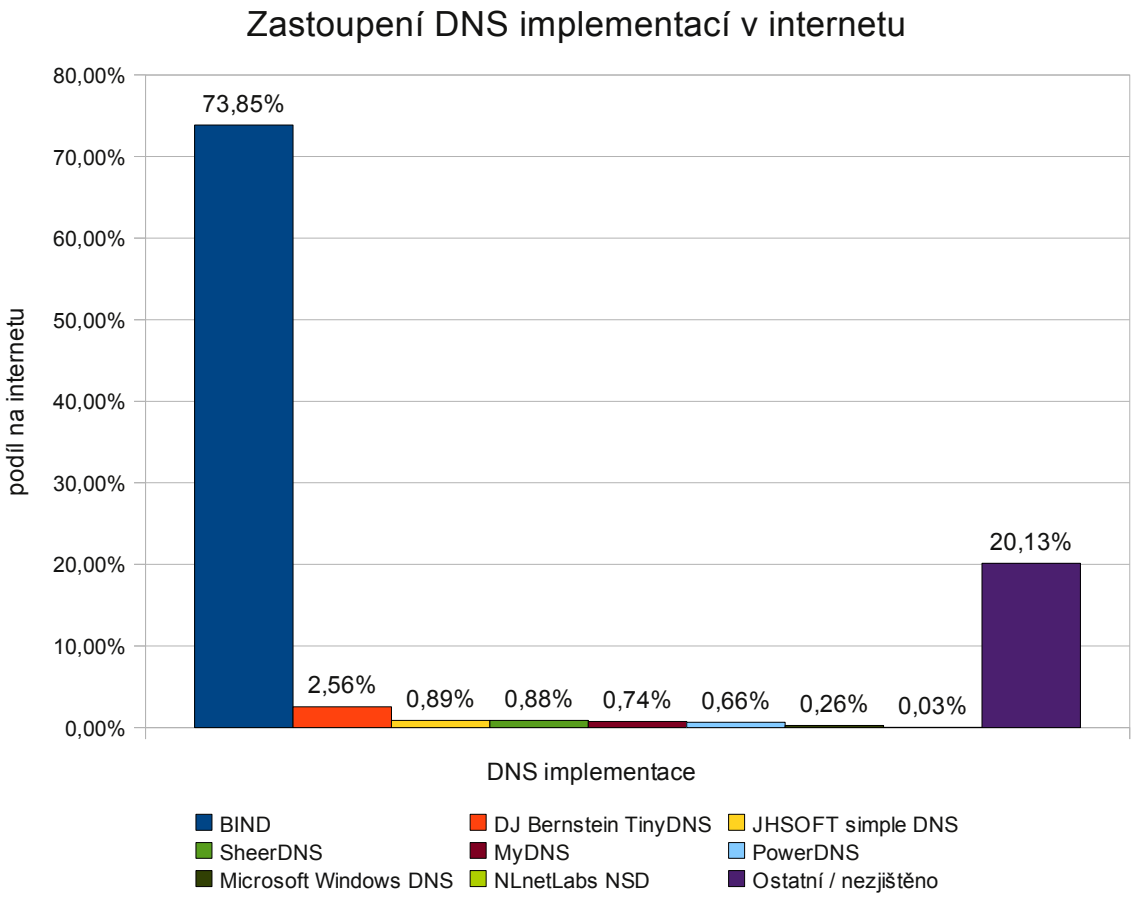
9 RRDtool je open source nástroj pro zaznamenávání numerických dat (ve formě „round robin databáze“), jednoduchou statistiku a grafické znázorňování dat v grafech.

10 Domovská stránka projektu je na adrese http://bind-dlz.sourceforge.net/.

4.3 Testované DNS implementace

Implementací DNS je velké množství, liší se mezi sebou velikostí, funkcemi, určením, bezpečností, licencí, historií atd. V podstatě spolu ale mohou různé implementace bez problému spolupracovat díky tomu, že jsou zde určité standardy a doporučení.

Pro to, abych byl schopný si vybrat vhodné implementace k testování, bylo třeba si udělat přehled o jejich podílu na internetu. Na internetu jsem našel řadu průzkumů s různou úrovní důvěryhodnosti, obsáhlosti a relevantnosti. Nakonec jsem se rozhodl vycházet z průzkumu DNS Survey[3], z jejich dat vytvořil následující ilustraci 4.2. Avšak rád bych ještě odkázal na další průzkum, a to „DNS Server Survey“[4].



Ilustrace 4.2: Zastoupení DNS implementací v internetu

Graf na ilustraci 4.2 znázorňuje procentuální podíl počtu instalací jednotlivých DNS implementací na autoritativních DNS serverech v internetu. Měření bylo provedeno v říjnu 2010 na 250 000 náhodně vybraných serverech.

Jak je z grafu patrné, BIND je suverénní jedničkou. Co se týče dalších pozic, ty jsou poměrně vyrovnané, zvláště když vezmeme do úvahy možnou chybu měření. Průzkumy se mezi sebou značně liší ve výsledcích, proto je nutné brát vše s rezervou.

Po konzultacích se sdružením CZ.NIC a pečlivém zvážení jsem pro další zkoumání vybral následující DNS implementace v daných verzích: BIND 9.7[11], NSD 3.2.5[12], PowerDNS 2.9.22[13] a MaraDNS 1.4.03[14]. Jedná se vždy o „open source“ projekty, díky tomu mohu analyzovat i jejich zdrojový kód.

Do testu nebyly zařazeny některé známé implementace s větším pokrytím trhu, a to kvůli následujícím důvodům. Microsoft DNS server, dostupný v serverových operačních systémech Windows, nebyl do testování začleněn kvůli svému zaměření na „malé“ (firemní a korporátní) DNS servery (co do velikosti a počtu zón). TinyDNS, SheerDNS MyDNS jsou neudržované a nadále již nevyvíjené implementace, nedá se u nich počítat s opravou chyb a podporou do budoucna. Zejména TinyDNS trpí díky své licenci, pod kterou je vydán. Posledním vyloučeným je JHSOFT simple DNS, který je pro operační systém Windows a komerční. Bylo by sice zajímavé mít i tuto implementaci začleněnu do testu, ale vzhledem ke komplikacím, které by to přineslo při měření, a nezajímavosti pro sdružení CZ.NIC, s ním nebudu nadále pracovat.

4.4 Analýza předpokladů

V této kapitole bych rád krátce představil jednotlivé implementace, a to se zaměřením na informace, které pomohou ve vytvoření představy o jejich současném a budoucím stavu. Takovými informacemi jsou například licence, pod jakou jsou vyvíjeny, kdo stojí za jejich vývojem, jaká je jejich historie, oblíbenost atd.

Znalost těchto informací je důležitá. Například na špatně zvolenou licenci doplatila jednu dobu velmi populární implementace djbdns[15] (TinyDNS) od D. J. Bernsteina. Implementace, za kterými stála pouze malá skupina lidí, na tom také nejsou nejlépe, například na to doplatila implementace MyDNS, která se dle vlastního průzkumu[16] dostala mezi šest nejpoužívanějších implementací. Ohledně bezpečnosti je zase jasné, že nikdo si na svůj server, který má zajišťovat vysokou dostupnost a služby platícím zákazníkům, nenasadí implementaci, která má za sebou hromadu kritických bezpečnostních chyb.

4.4.1 BIND

Plným jménem „Berkeley Internet Domain Name“ je DNS server s historií od počátku osmdesátých let. V současné době stojí za vývojem BINDu nezisková organizace „Internet Systems Consorcium (ISC)“. BIND je vyvíjen pod BSD licencí.

Současná stabilní verze BINDu je 9.7. Od roku 2009 se již pracuje na nové verzi s číslem 10.x, 19. dubna 2010 byla vypuštěna první testovací verze, avšak vypuštění ostré verze si ještě vyžádá několik roků.

11 Domovská stránka implementace BIND je http://www.isc.org/software/bind.

12 Domovská stránka implementace NSD je http://www.nlnetlabs.nl/projects/nsd/.

13 Domovská stránka implementace PowerDNS je http://www.powerdns.com/product/powerdns-server.aspx.

14 Domovská stránka implementace MaraDNS je http://www.maradns.org/.

15 Djbdns http://cr.yp.to/djbdns.html.

16 MyDNS server survey http://mydns.bboy.net/survey/.

Dle stránky ISC BIND Security Advisories[17] bylo, za období 8.11.1999 – 19.1.2010, v BINDU nalezeno 34 bezpečnostních chyb, z toho jich bylo 19 s hodnocením high, critical nebo serious. BIND je přitom oproti ostatním v „nevýhodě“, kvůli svému téměř dominantnímu postavení je o jeho prolomení velký zájem.

BIND je nejrozšířenějším DNS serverem, jak podle názoru odborníků, tak i dle ilustrace 4.2. BIND je také nasazen na většině současných DNS kořenových serverech[18].

4.4.2 NSD

Plným jménem „Name Server Daemon“ je DNS server vyvíjený nizozemskou skupinou (nejspíše nezisková organizace) NLnet Labs pod licencí BSD. Podle záznamů v SVN tipuji začátek vývoje na rok 2001. Během této krátké historie se NSD dostal již do své třetí generační (major) verze. Opět podle záznamů v SVN tipuji že, ve výše zmíněných výkonnostních testech byla testována teprve druhá řada (2.x.x).

Co se týče různých chyb, tak je na tom poměrně dobře. Dle oficiálního buglistu[19] jich bylo nalezeno a opraveno 51 skrze všechny verze NSD. Z toho pouze jednu chybu uvádí Nlnet Labs jako kritickou pro bezpečnost. Jedná se o chybu typu „buffer overflow“, která umožňuje shodit NSD.

Spolu s BINDem je to jediný DNS server, který je nasazen i na kořenových DNS serverech.

4.4.3 PowerDNS

PowerDNS je jmenný server, který původně vznikal jako komerční projekt, avšak kvůli nezdarům se od těchto úmyslů nakonec upustilo. V současné době je PowerDNS zdarma pod GNU/GPL v2. Vývoj, včetně komunity, zajišťuje stejnojmenná společnost, která k serveru poskytuje komerční podporu. Opět dle SVN odhaduji počátek vývoje PowerDNS na rok 2001.

I když PowerDNS je propagován jako rychlý a výkonný server (dle slov autorů), předchozí testy to zatím nedokazují. Jeho hlavní plus a důvod obliby bych tedy viděl v množství backendů. V současné době podporuje jako backend většinu standardně používaných databází, LDAP, standardní BIND soubory aj.

Co se týče chyb, vypadá to s PowerDNS poměrně dobře. Vývojový tým všechny chyby spojené s bezpečností zveřejňuje na stránkách s dokumentací[20], z toho polovina chyb se týká pouze rekurzivní části PowerDNS (ta nás v tomto testu nezajímá, navíc je jako samostatný program).

4.4.4 MaraDNS

MaraDNS je implementace autoritativního i rekurzivního DNS serveru, vydávaná pod licencí BSD. Server je aktivně vyvíjen od roku 2001. Autor o svém serveru tvrdí, že nejvyšší jeho prioritou je bezpečný kód.

Na rozdíl od všech ostatních se jedná o DNS server s nejistou budoucností. Důvodů k mému názoru mám více. Jedná se totiž o „one man show“ a i přes BSD licenci to nevypadá, že by okolo něj byla větší komunita. Přestože autor mluví o bezpečnosti, tak v jeho kódu bylo odhaleno 12

17 K nalezení na http://www.isc.org/advisories/bind.

18 Dle informací dostupných na Wikipedii a webových stránkách jednotlivých kořenových serverů

19 NSD buglist je dostupný na adrese http://www.nlnetlabs.nl/labs/bugs/.

20 Seznam bezpečnostních chyb PowerDNS http://doc.powerdns.com/security-policy.html.

bezpečnostních chyb[21]. Sám autor se snaží všechny chyby komentovat a „uvádět je na pravou míru“ (někdy to působí i jako ututlávání). Autor sám neví, zda bude software vyvíjet i nadále, například na Wikipedii[22] se uvádí, že autor sekne s vývojem jakmile vypustí verzi 2.0. Avšak 1. dubna 2010 vyšla na jeho blogu nová roadmapa k vývoji nových verzí, kde autor píše: *„Today I have decided to revise my MaraDNS roadmap. I have realized that I have an intrinsic need to develop open-source software, and that this need is so strong it doesn't matter if I get paid for my work. That in mind, I will quit my job and have time to develop all of the features people wish.“[5].* Autor dále v článku pokračuje tím, že by chtěl každý rok následně vydávat novou hlavní verzi.

Avšak přes všechny výše zmíněné nevýhody jsem se rozhodl tuto implementaci zařadit do testu. Přiměly mne k tomu dále zmíněné důvody. Software jsem ještě neviděl v žádném testu. Software si již získal určitou oblibu a počet produkčních nasazení. Autor vývojem svého software žije. Jeho vývoji již věnoval přes devět let, čímž by měla být vyloučena možnost pouze krátkodobého vzplanutí pro věc. Dle jeho obhajoby MaraDNS[23] je vidět, že v něm vidí budoucnost a věří mu.

4.4.5 Vyhodnocení

Všechny zde popisované implementace jsou pod vhodnými licencemi, mají relativně dlouhou historii a přiměřený počet bezpečnostních incidentů. Myslím si, že výběr kandidátů pro testování mohu označit jako vydařený a zajímavý pro budoucí praktické testování.

Pokud bych tu však měl krátce shrnout můj názor na každou distribuci, pak by vypadal takto. BIND je v internetu pomalu synonymem pro DNS. Jedná se o DNS implementaci s nejdelší tradicí a nejjistější budoucností stálého vývoje.

Jestli někdo sesadí BIND ze svého trůnu nejvíce používané implementace, tak to bude s největší pravděpodobnější právě NSD. Využití PowerDNS vidím hlavně u drobných poskytovatelů webhostingových služeb, jelikož už velmi dlouho přináší databázové backendy, které jsou vhodné pro jednoduché psaní grafických administračních rozhraní. Na druhou stranu ony databázové backendy jsou rozhodně jeho výkonnostní brzdou. U MaraDNS si nejsem jist jeho budoucností z důvodu malé komunity. Jsem ovšem zvědavý na praktické testy, které kdyby dopadly dobře, mohly by moje obavy zmírnit.

Tato krátká kapitola s představením jednotlivých distribucí a jejich drobným rozborem, potvrdila můj výběr implementací k testování jako povedený.

4.5 Modelová konfigurace

Pro možnost reprodukce testů či co nejlepší interpretace výsledků testů je třeba znát nastavení testů a testovaného prostředí (jednotlivých implementací).

21 Seznam bezpečnostních chyb MaraDNS http://maradns.org/security.html.

22 O MaraDNS na Wikipedii http://en.wikipedia.org/wiki/MaraDNS.

23 MaraDNS Advocacy http://www.maradns.org/advocacy.html.

4.5.1 HW a OS

Znalost použitého HW a OS je důležitá například pro srovnávání se stejnými testy provedenými na jiném HW, proto zde uvádím veškeré údaje, které jsou dle mne relevantní.

Důležité je také vzájemné propojení obou počítačů. Aby testy byly co nejméně ovlivněné případnými dalšími prvky, byly počítače při testu propojeny napřímo s pomocí kroucené dvojlinky.

4.5.1.1 Testovaný počítač

Jednotlivé implementace budou testovány na dedikovaném PC. Parametry PC jsou:

- Intel (R) Celeron (R) CPU 420 @ 1.60GHz
- 2x512MB RAM, DDR2 800MHz, Dual channel
- 160GB SATA HDD, 7200RPM (HDS722516VLSA80)
- 100Mbitová síťová karta.

Jako operační systém, na kterém poběží jednotlivé implementace, bude použit GNU/Linux Gentoo. Tento operační systém byl zvolen záměrně, protože:

- vše je kompilováno přímo ze zdrojových kódů,
- nabízí nastavení přepínačů překladače,
- kompiluje programy s volitelnými součástmi (dá se měnit již krok „./configure“),
- umožňuje otestovat více verzí závislostí (knihoven, databází, interpretrů),
- mám s ním mnohaleté zkušenosti.

Jiné specifické požadavky na HW nejsou, protože to není důležité ani směrodatné. Dalo by se i říci, že čím rychlejší testovaný počítač, tím hůře se bude měřit systémové zatížení (počítač bude velmi těžké pořádně vytížit).

4.5.1.2 Počítač generující zátěž

Pro zatěžování DNS implementací jsem použil svůj soukromý notebook, který je dostatečně výkoný aby testovací počítač přiměřeně vytížil.

4.5.2 Záznamy DNS

V této kapitole popisuji jak, proč a z čeho se skládají zónové soubory obsluhované implementacemi DNS a vzhled konfiguračního souboru s dotazy DNS.

4.5.2.1 Zónové záznamy

Tato část je pro všechny implementace společná. Jedná se o to, pro jaké informace (záznamy DNS) budou tyto implementace vystupovat jako autoritativní servery (co budou obsluhovat). Jelikož je tato bakalářská práce vytvářena ve spolupráci se sdružením CZ.NIC, bude se modelová konfigurace podobat jejich.

CZ.NIC je správcem domény TLD CZ a 0.2.4.e164.arpa. Z toho plyne, že jejich jmenné servery jsou autoritativními servery pro dvě zóny (domény) a každá zóna obsahuje obrovské množství záznamů. Já jsem v modelové konfiguraci použil pouze jednu zónu, a to z důvodu, že obsluhovat dvě obrovské zóny by nemusel starší HW na testovacím PC zvládnout.

Když jsem vymýšlel, jak vytvořit modelovou zónu CZ pro testování co nejvěrohodnější v porovnání se skutečnou zónou CZ, tak jsem zvažoval i možnost získat skutečnou zónu kvůli „chybě“ v návrhu DNSSECu. Získání zóny pomocí záznamů typu NSEC jsem brzo zavrhl, jelikož jsem zjistil, že by to vyžadovalo delší dobu skriptování s nejistým výsledkem. Rozhodl jsem se, že si tedy vygeneruji svou zónu, která by se měla podobat té originální.

Důvod pro to, aby se zóna podobala té originální, je jednoduchý. Náhodně vygenerované DNS záznamy, nesmyslná „slova“ s rovnoměrným výskytem znaků, by nereflektovaly rozložení a výskyt písmen v českém jazyce.

Doménové jméno	**Typ doménového jména**
jiri.cz	jednoduché doménové jméno
zlatuska.cz	jednoduché doménové jméno
jiri-zlatuska.cz	složené doménové jméno
zlatuskajiri.cz	složené doménové jméno
xn--ji-oja54b.cz	IDN doménové jméno
xn--zlatuka-uqb.cz	IDN doménové jméno

Tabulka 4.2: Ukázka vygenerovaných doménových jmen.

Vzal jsem tedy jeden český slovník, nad kterým jsem provedl transliteraci a odstranil duplikáty. Zůstalo mi přes 150 tisíc slov. Jejich kombinováním jsem utvořil dalších 300 tisíc „dvojslov“, která byla náhodně oddělená pomlčkou nebo byla bez oddělovače. Na závěr jsem ještě vybral 100 tisíc slov s diakritikou[24] (IDN – Intenationalized domain name) a převedl je do ACE[25] (ASCII Compatible Encoding) formátu. Ukázka vygenerovaných doménových jmen je v tabulce 4.2.

DNS Záznam				**Komentář**
petr.cz	IN	NS	ns1.petr.cz	Delegovaná doména s NS záznamy ukazujícími dovnitř sebe sama.
petr.cz	IN	NS	ns2.petr.cz	
ns1.petr.cz	IN	A	74.74.74.74	Glue záznamy.
ns2.petr.cz	IN	A	74.74.74.75	
martin.cz	IN	NS	ns1.petr.cz	Delegovaná doména s NS záznamy ukazujícími do jiné domény.
martin.cz	IN	NS	ns2.petr.cz	

Tabulka 4.3: Ukázka zónového souboru.

Pro každé takto vygenerované doménové jméno bylo následně třeba vytvořit delegaci. Vytvoření delegace znamená, že se v zónovém souboru vytvoří odkaz na jmenné servery autoritativní pro tuto doménu. Náhodně jsem proto rozdělil doménová jména na dvě poloviny. První polovina měla NS záznamy s odkazem dovnitř své doménové zóny, a proto bylo potřeba vytvořit k těmto jmenným serverům i „glue“ záznamy (záznamy typu A sdělující IP adresu onoho jmenného serveru). Druhá polovina měla NS záznamy odkazující na jmenné servery v jiných doménách. Pro lepší pochopení, je možno nahlédnout na ukázku v tabulce 4.3.

Jelikož je již delší dobu zóna CZ zabezpečena pomocí technologie DNSSEC, udělal jsem to samé s mojí zónou. Podepsání zóny zásadně ovlivňuje modelovou konfiguraci, jelikož DNSSEC záznamy jsou velmi objemné (zvětší zónu o více jak 100%). CZ.NIC dle svého „Provozního manuálu

24 Mezinárodní doménová jména označovaná zkratkou IDN (Internationalized domain name) více například na http://www.háčkyčárky.cz.

25 I když v současné době v CZ doméně není „oficiálně“ zavedena podpora IDN (Internationalized Domain Name), tak v jiných TLD doménách již je, proto jsem se rozhodl dát do modelové zóny IDN záznamy. IDN je již zavedeno například v pro nás „vlastní“ TLD EU.

k používání technologie DNSSEC“[26] používá délku klíče KSK (key signing key) 2048 bitů a délku klíče ZSK (zone signing key) 1024 bitů. Výsledná velikost podepsané zóny byla nakonec příliš velká, a proto jsem ji musel trochu zmenšit. Složení a vlastnosti jednotlivých zón lze najít v přehledné tabulce 4.4.

Zóna	**Bez DNSSECu**	**S DNSSECem**
Jednoduchých doménových jmen	150 000	150 000
Složených doménových jmen	300 000	100 000
IDN doménových jmen	100 000	100 000
Celkem delegovaných domén	550 000	350 000
NS záznamů	1 100 000	700 000
Glue záznamů	550 000	350 000
DS záznamů	0	70 000
NSEC a RRSIG záznamy	0	770 000
Celkem resource záznamů	1 650 000	1 890 000
Velikost zóny v MB	81,00	179,60

Tabulka 4.4: Parametry zón spravovaných DNS implementacemi

4.5.2.2 Konfigurační soubor s dotazy DNS

Věřím, že z názvu nemusí být hned jasné, o co se jedná, bohužel jsem však nenašel výstižnější název. Pod názvem konfigurační soubor s dotazy DNS se skrývá soubor obsahující informace, na jaké záznamy DNS (doménové jméno a typ) se má klient ptát. Dále tento soubor ještě obsahuje řídící příkazy pro program Queryperf.

Tento soubor je předán programu Queryperf jako parametr. Queryperf postupně od začátku souboru vybírá záznamy DNS, na které se ptát, a posílá dotazy na DNS server. Tento soubor je zaveden proto, aby byla každá implementace podrobena identickému sledu dotazů DNS a výsledky testů tak byly mezi sebou co nejvíce porovnatelné.

Konfigurační soubor jsem sestavil tak, že 96% jeho obsahu jsou dotazy na existující delegované domény v rámci CZ zóny. Zbylá 4% jsou dotazy na neexistující doménové záznamy v rámci CZ zóny. Dotazy směřující do neexistujících (neobsluhovaných) zón nejsou uvažovány, jelikož se u čistých TLD nepředpokládá, že by na ně bylo delegováno, něco co nespravují[27].

26 Ke stažení na adrese http://www.dnssec.cz/files/nic/doc/Provozni_manual_DNSSEC_201001_final.pdf.

27 Při různých pokusných měřeních se mi však podařilo zjistit, že dotazy mimo spravované zóny jsou obsluhovány mnohem rychleji (až v řádu desítek procent).

Obsah souboru	Komentář
#qps 100	Řídící příkaz pro Queryperf, nastavuje počet dotazů za sekundu na sto.
web.abatyse.cz A	Říká programu, aby se zeptal na IP adresu serveru web.abatyse.cz.
pes.les.cz MX	Říká programu, aby se zeptal na adresu poštovního serveru pro doménu pes.les.cz .
ns1.louka.cz A	Říká programu, aby se zeptal na IP adresu serveru ns1.louka.cz .
mail.posta.cz NS	Říká programu, aby se zeptal na IP adresu jmenného serveru pro doménu mail.posta.cz .

Tabulka 4.5: Ukázka konfiguračního souboru s dotazy DNS.

4.5.3 Nastavení implementací DNS serverů pro testování

Následuje výpis nastavení jednotlivých implementací. Pokud neuvedu jinak, tak se předpokládá, že vše zůstalo nastavené na výchozích hodnotách.

- U všech implementací bylo vypnuto logování. Toto rozhodnutí jsem provedl po konzultaci s lidmi z praxe. Zkoušel jsem i rychlý test s logováním a bez. Výsledkem bylo zjištění, že logování zpomaluje výkon DNS implementací zhruba o polovinu.
- Všechny implementace, až na MaraDNS, používají totožné zónové soubory, jelikož podporují stejnou strukturu.
- Implementace BIND a NSD používají již předkompilované zóny (umožňuje rychlejší načítání). Pokud by i jiné implementace toto uměly, bylo by to u nich použito.
- MaraDNS má jako jediná implementace jiný formát zónových souborů, a tak pro ni musel být zónový soubor přeformátován.
- Implementace MaraDNS obsahovala konstrukční omezení u velikosti zóny. V kódu je pevně nastaven limit 100000 záznamů na jednu zónu. Modelové zóny však mají více jak 15krát tolik. Provedl jsem tedy analýzu zdrojového kódu a tyto limity opravil. Bohužel se mi ale kvůli nedostatku paměti podařilo načíst pouze 1,4 miliónu záznamů, a tak je MaraDNS implementace v testech mírně zvýhodněna.
- PowerDNS byl nastaven na použití s BIND backend a vypnuta packet cache. Vypnutí packet cache bylo provedeno na základě výkonnostních doporučení samotných tvůrců PowerDNS[28].

4.6 Statické testy

V této kapitole jsou prezentovány implementace a výsledky „statických“ testů. Výsledky těchto testů jsou vhodné pro rychlé porovnání implementací mezi sebou, avšak o jejich chování toho příliš neprozradí.

28 Viz Authoritative server performance http://doc.powerdns.com/performance.html.

4.6.1 Doba startu

Tento test ukazuje, jak dlouho od spuštění trvá, než je server schopen obsluhovat DNS dotazy. K měření byl použit nástroj „timeDnsRefresh.pl“. Rychlost startu DNS serveru je důležitá v případě pádu DNS implementace nebo celého stroje. Čím víc je obsluhovaných zón a čím jsou větší, tím déle trvá start.

Na testovacím stroji byla spuštěna implementace DNS a na zatěžovacím PC tento nástroj (oboje současně). Naměřená hodnota byla zaznamenána a testovací počítač byl restartován. Toto se provedlo pro každou implementaci třikrát. Restarty bylo zaručeno, že žádná data nezůstanou v paměti a naměřené hodnoty nebudou zkreslovány.

Výsledky jsou zaznamenány v tabulce 4.6. Implementace BIND a NSD byly testovány na obou zónách, jelikož jako jediné podporují DNSSEC. V testu je jednoznačně nejlepší implementace NSD a nejhůře je na tom PowerDNS.

Zajímavý je paradox u NSD, kdy načítání DNSSEC zóny, i když je větší, trvalo kratší dobu. Dokážu si to vysvětlit tak, že u záznamů typu RRSIG a NSEC se provádí méně validací. Dalším vysvětlením by mohlo být, že se mu rychleji sestavuje „provozní seznam“, když má více záznamů patřících ke stejnému doménovému jménu.

	Doba startu [s]				Směrodatná
Implementace / měření	**1.**	**2.**	**3.**	**Průměr**	**odchylka**
PowerDNS	31	31	33	31,67	1,15
BIND	13	13	13	13,00	0,00
BIND (DNSSEC)	14	13	14	13,67	0,58
MaraDNS	26	25	25	25,33	0,58
NSD	10	9	10	9,67	0,58
NSD (DNSSEC)	7	7	7	7,00	0,00

Tabulka 4.6: Výsledky měření doby startu jednotlivých DNS implementací.

4.6.2 Maximální zátěž

Jedná se o test prováděný pro zjištění maximálního možného počtu dotazů za sekundu, který je schopna implementace DNS obsloužit.

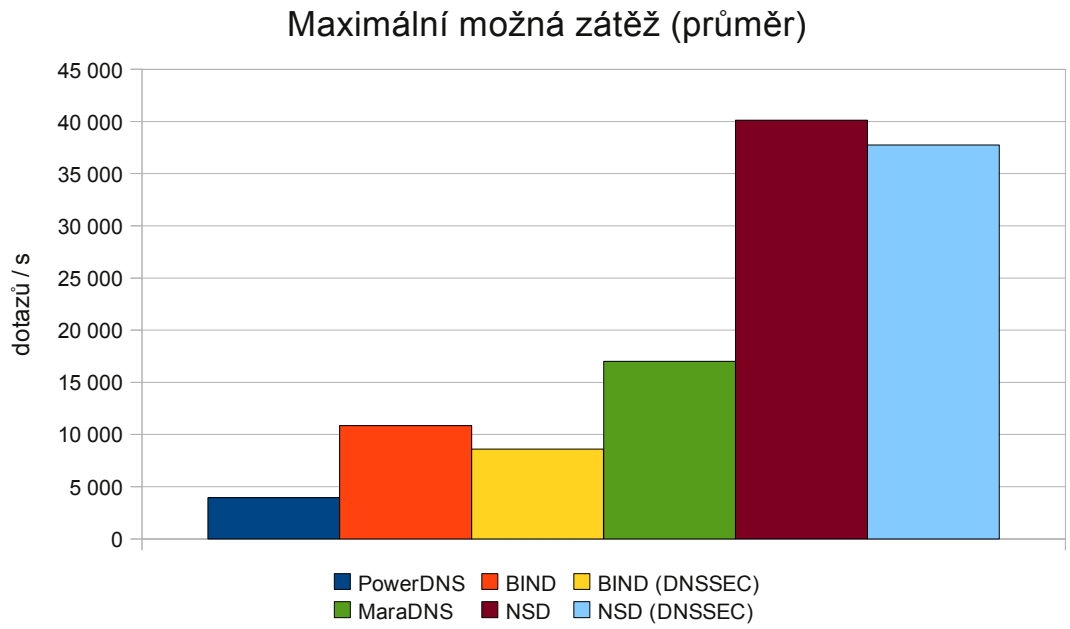
Test je prováděn s pomocí nástroje Queryperf a konfiguračního souboru s dotazy DNS. Test probíhá tak, že Queryperf čte dotazy z konfiguračního souboru a zasílá je co nejrychleji na DNS server, vždy ale tak, aby byl dodržen parametr „maximum number of outstanding queries“. Tento parametr je funkcí podobný například TCP parametru Window. Laicky řečeno, určuje, kolik může být maximálně v každou chvíli dotazů, na které ještě nepřišla odpověď. Pokud je této hranice dosaženo, pak je posílání dalších dotazů pozastaveno, dokud se počet dotazů bez odpovědi nesníží. Ke snížení počtu dotazů bez odpovědi dochází při dvou událostech, pokud dorazí odpověď na dosud nezodpovězený dotaz, nebo uplyne-li doba (timeout), po kterou se čeká na odpověď.

Dalšími parametry testu byly doba běhu nastavená na pět minut a velikost parametru „maximum of outstanding queries“ nastavená na sto. U zóny s technologií DNSSEC byla velikost parametru „maximum of outstanding queries“ snížena na dvacet. Ke snížení jsem přistoupil potom, co docházelo k častým ztrátám odpovědí z důvodu velkého zpoždění (v řádu desítek sekund). Objevit příčinu tohoto problému se mi nepodařilo, ale je pravděpodobné, že se jedná o chybu v programu

Queryperf.. Tento test byl pro každou implementaci proveden třikrát a naměřené výsledky jsou zaneseny v tabulce 4.7. Z naměřených čísel a velikostí směrodatných odchylek lze usoudit, že již samotná délka běhu testu se postarala o „stabilní“ výsledky. pro jednodušší porovnání jsou výsledky znázorněny i v grafu 4.3.

	Maximální možná zátěž [dotazů/s]				Směrodatná odchylka
Implementace / měření	**1.**	**2.**	**3.**	**Průměr**	
PowerDNS	3 941	3 955	3 948	3 948	7,00
BIND	10 855	10 842	10 831	10 843	12,01
BIND (DNSSEC)	8 600	8 622	8 598	8 607	13,32
MaraDNS	17 015	16 989	16 993	16 999	14,00
NSD	40 122	40 151	40 135	40 136	14,53
NSD (DNSSEC)	37 746	37 767	37 755	37 756	10,54

Tabulka 4.7: Naměřená maximální zátěž pro jednotlivé implementace.



Ilustrace 4.3: Maximální naměřená zátěž jednotlivých implementací zanesená do grafu.

Z uvedených výsledků je jasným vítězem implementace NSD. Překvapením je i implementace MaraDNS. Zklamáním je však PowerDNS, která je oproti vítězi desetkrát pomalejší.

Rád bych ještě upozornil, že každý, kdo provozuje autoritativní DNS, by neměl nikdy dosáhnout maximální možné zátěže jeho implementace měřené na jeho stroji a zátěž jeho DNS serveru by měla být i ve špičce nižší. Spojení „neměl dosáhnout“ není myšleno způsobem „že to nelze“, ale že by to neměl dopustit. Z mých drobných měření totiž vyplývá, že se implementace po překročení svého maxima začínají chovat nevyzpytatelně a jejich výkonost se už prudce snižuje a dochází k exponenciálnímu zvyšování nezodpovězených dotazů z důvodu timeoutu.

4.6.3 Vyhodnocení

Dle výsledků bych řekl, že testy se povedly a jsou přínosné. Nejlépe dopadla ve statických testech implementace NSD, která byla nejlepší ve všech testech. Naopak implementace PowerDNS dopadla nejhůře a ve všech testech byla nejhorší.

4.7 Behaviorální test

Účelem tohoto testu je udělat charakteristiku závislosti zdrojů (CPU, disk, paměti atd.) na zátěži. Při tvoření této charakteristiky kladu důraz na to, aby nastavení testů a prostředí odpovídalo co nejvíce skutečnému prostředí a skutečnému provozu[29].

Při behaviorálních testech je použit nástroj Queryeperf, mnou vytvořené měřící skripty a moduly monitorovacího systému Munin.

Testovací PC je před každým testem restartováno a je na něm spuštěna pouze testovaná implementace DNS. Vše ostatní, co není pro její běh potřeba, je vypnuto (resp. po restartu ani není povoleno ke startu). Následně jsou nastaveny parametry měřícího skriptu. Na tomto PC se měří CPU, Disk I/O, operační paměť a počet procesů s vlákny.

Na PC generujícím zátěž (resp. mém NB) jsou nastaveny parametry[30] měřícího a zatěžovacího skriptu (ve skutečnosti se jedná o jeden skript). Na tomto PC se měří počet dotazů za sekundu a zpoždění.

Jakmile je benchmark započat, začne zatěžovací PC zasílat dotazy z konfiguračního souboru (toho s dotazy DNS) rychlostí 100 dotazů za sekundu. Každých pět sekund je na obou počítačích provedeno měření aktuálního stavu a zatěžovací PC zrychlí zasílání DNS dotazů o 100 dotazů za sekundu. Takto to pokračuje, dokud není dosaženo předem stanovené doby běhu. Doba běhu se stanovuje jednoduchým výpočtem tak, aby skripty proměřily DNS implementaci až do její maximální zatížitelnosti (změřené v testu „maximální zátěž“). U testů nad nepodepsanou zónou byl nastaven parametr „maximum number of outstanding queries“ na 100 a nad podepsanou zónou na 20.

Implementace	Délka měření [s]	Konečný počet dotazů za sekundu	Počet provedených měření
BIND 9.7	530	10600	106
BIND 9.7 DNSSEC	500	10000	100
MaraDNS 1.4.03	825	16500	165
NSD 3.2.5	2 015	40300	403
NSD 3.2.5 DNSSEC	1 950	39000	390
PowerDNS 2.9.22	225	4500	45

Tabulka 4.8: Parametry provedených testů pro jednotlivé implementace

29 Skutečným provozem je v tomto případě myšlen provoz z pohledu registrátora TLD či jiného správce DNS serveru s obrovským počtem zón a záznamů v nich.

30 Nastavuje se jméno testované implementace, aby mohlo být vloženo do grafů, a také doba běhu skriptu. Doba běhu skriptu je vypočítána ze znalosti maximální možné zátěže dané DNS implementace a charakteristiky logu s DNS dotazy.

4.7.1 Čtení grafů

Výsledné grafy jsou trochu složitější ke čtení, a proto bych zde rád napsal pár rad, jak je číst. I když všechny grafy zachycují závislost některého ukazatele na vzrůstajícím počtu dotazů DNS, tak veličinou osy X je čas. Pro převod času na počet dotazů za sekundu je třeba nahlédnout do grafu závislosti latence na počtu dotazů za sekundu dané implementace, kde je na druhé ose Y vynesena hodnota počtu dotazů za sekundu pro každý časový okamžik. Je také možné převést čas na počet dotazů za sekundu pomocí vzorce uvedeného níže (1). Při použití vzorce však interpretace nebude přesná, jelikož při testech implementace ke konci testu často nestíhaly a skutečný počet dotazů za sekundu se liší od požadovaného počtu dotazů za sekundu. O tomto se ještě rozepisuji v kapitole s výsledky měření zpoždění.

$$počet\ dotazů\ za\ sekundu = zaokrouhlení\ dolů\ na\ celá\ čísla(počet\ sekund/5) * 100 + 100 \qquad (1)$$

Na ose X je uveden čas místo počtu dotazů za sekundu kvůli funkcionálnímu omezení nástroje RRDTool. Tento nástroj totiž neumožňuje nastavit veličinu osy X na jinou veličinu než čas.

Je třeba také dát si pozor na rozsah grafů. Jelikož zde mají všechny grafy stejnou velikost, může to působit, že mají i stejný rozsah osy X, což není pravda. Rozsah osy X pro jednotlivé implementace je přehledně srovnán v tabulce 4.8, ve sloupci délka měření.

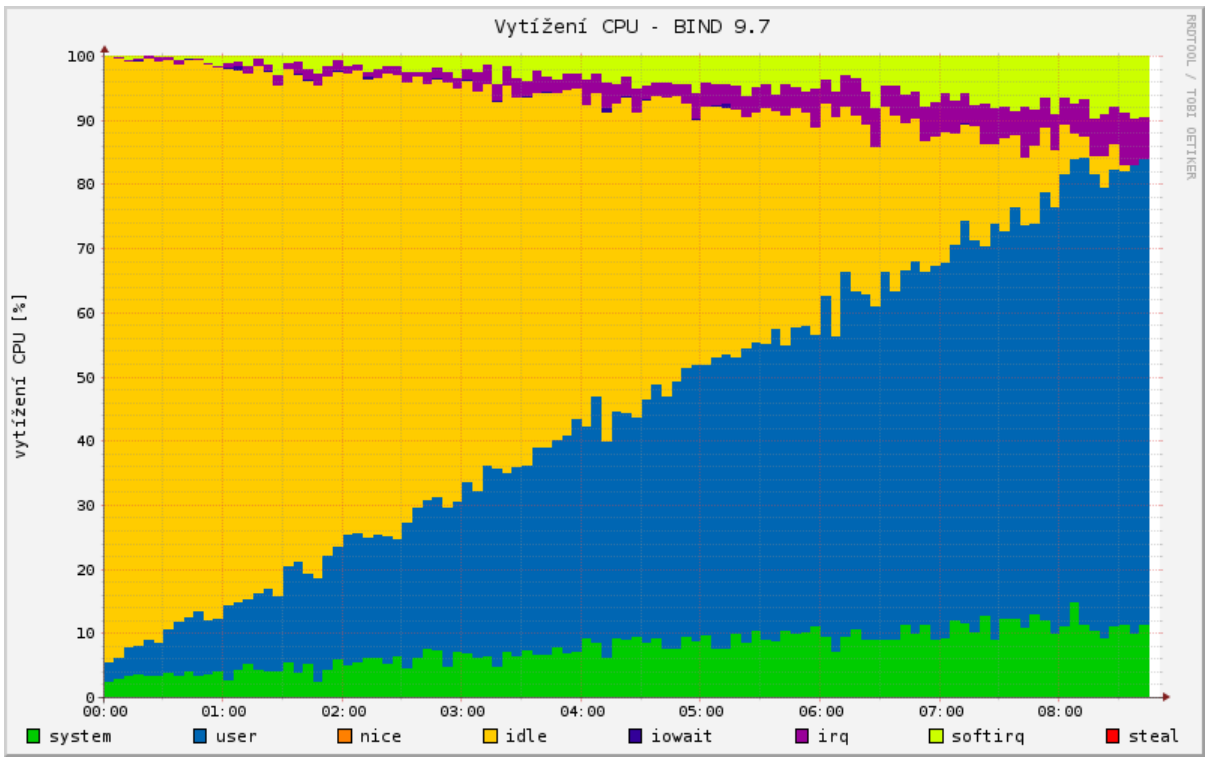
4.7.2 Využití CPU

Měření využití CPU probíhalo pomocí modulu, který si bral tuto hodnotu ze souboru „/proc/stat“. Jedná se o kumulativní informaci, takže naměřená hodnota se vždy rovnala průměru za interval měření. Šikovné a užitečné je, že v grafech je vidět, jak procesor strávil svůj výpočetní čas.

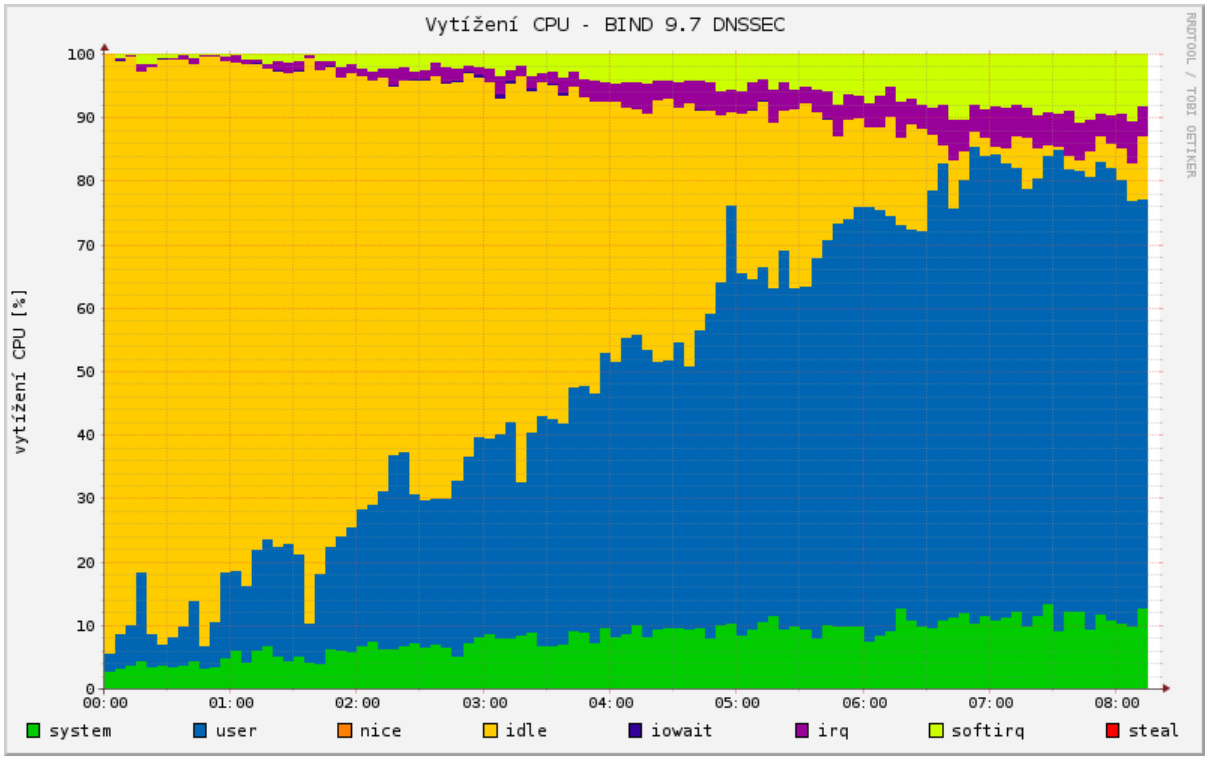
Graf vytížení CPU pro implementaci BIND (vyobrazený na ilustraci 4.4) je dle představ. Využití CPU roste spolu se zátěží lineárně a neobjevují se v něm větší výkyvy. Na konci měření je již CPU plně vytíženo, což byl náš záměr.

Naměřené vytížení CPU pro BIND se zapnutou technologií DNSSEC (ilustrace 4.5) již tak ideální není. Graf obsahuje různé výkyvy, růst již není tak lineární a CPU se nám nepodařilo vytížit na sto procent. Zvláštní je, že CPU není v závěru vytíženo na sto procent i přesto, že již implementace nezvládala více dotazů za sekundu. Za tak rozdílný vzhled výsledků od výsledků implementace BIND bez technologie DNSSEC může nejspíše rozdílná velikost parametru „maximum number of outstanding queries“.

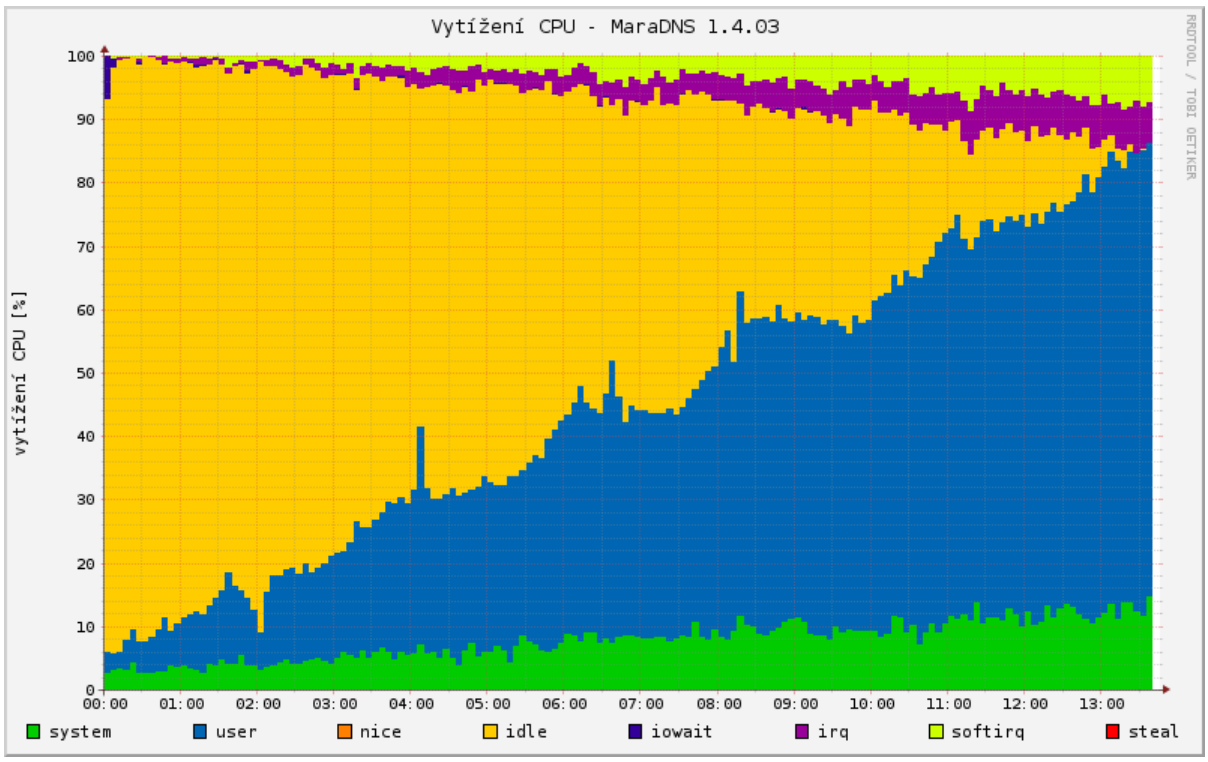
Charakteristika pro implementaci MaraDNS (ilustrace 4.6) je zajímavá ve své „houpavosti“, kde vytížení CPU v uživatelském módu nemá lineární závislost na počtu dotazů za sekundu. Graf neobsahuje téměř žádné nenadálé výkyvy a implementace na konci měření vykazuje plné využití CPU. Také je zde vidět, že CPU u této implementace tráví o polovinu méně času obsluhou softwarových přerušení než u implementace BIND.



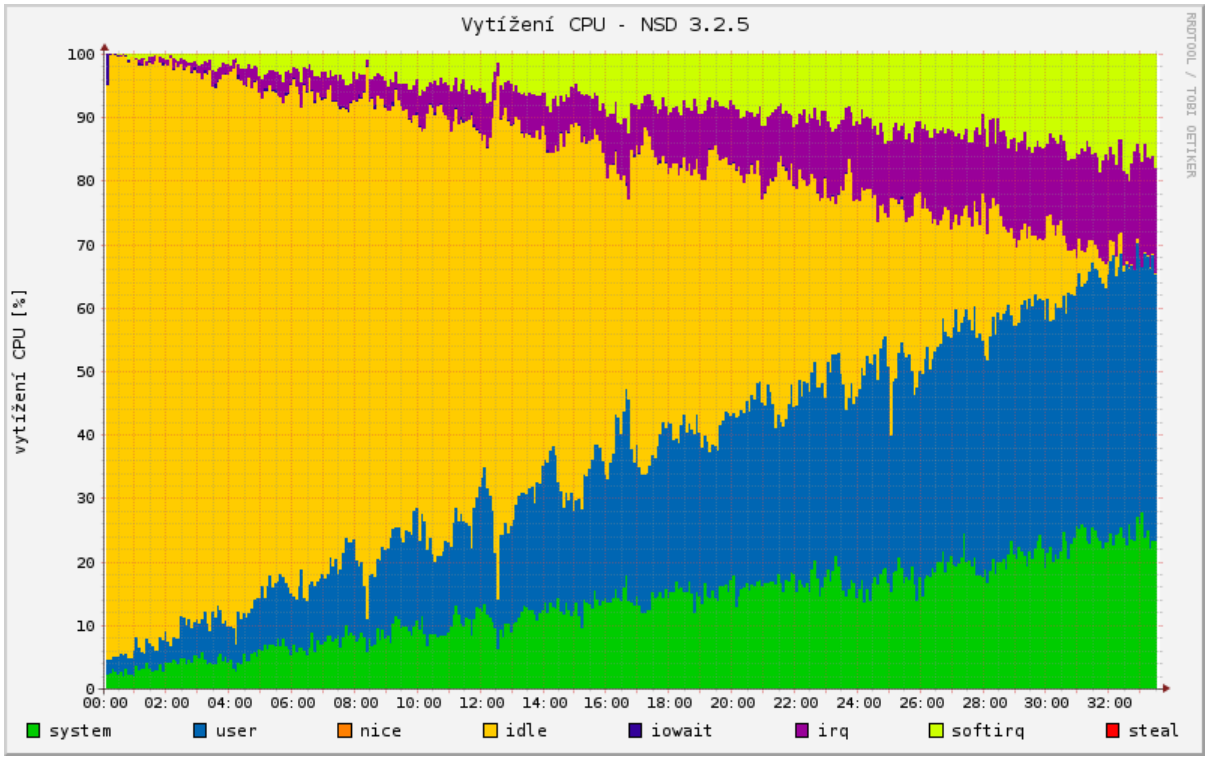
Ilustrace 4.4: Vyžití CPU – BIND 9.7



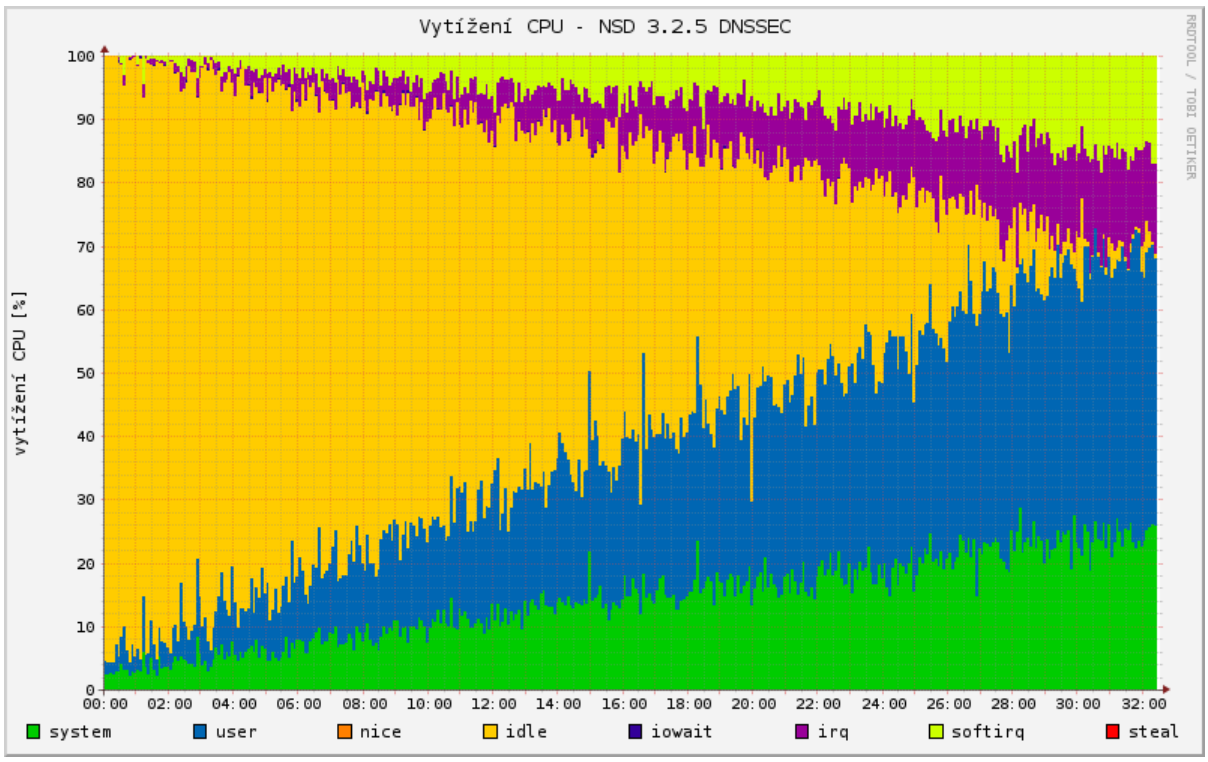
Ilustrace 4.5: Vyžití CPU – BIND 9.7 DNSSEC



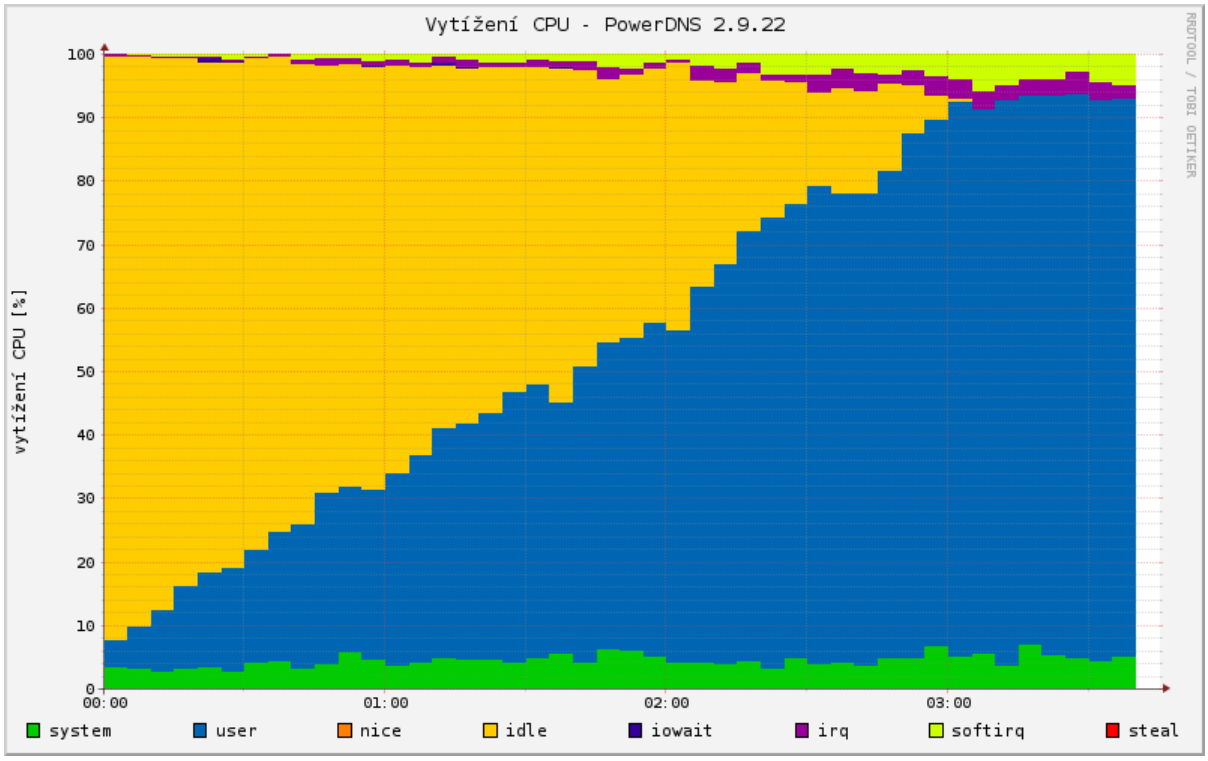
Ilustrace 4.6: Využití CPU – MaraDNS 1.4.03



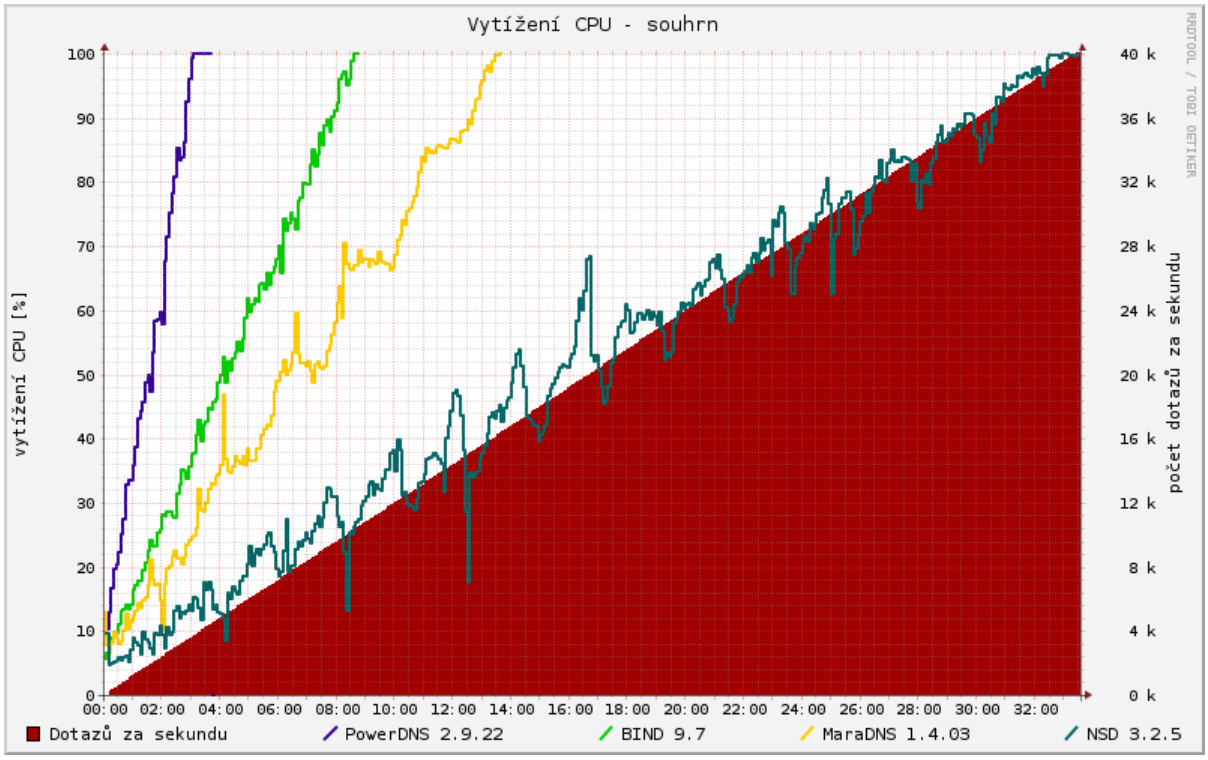
Ilustrace 4.7: Využití CPU – NSD 3.2.5



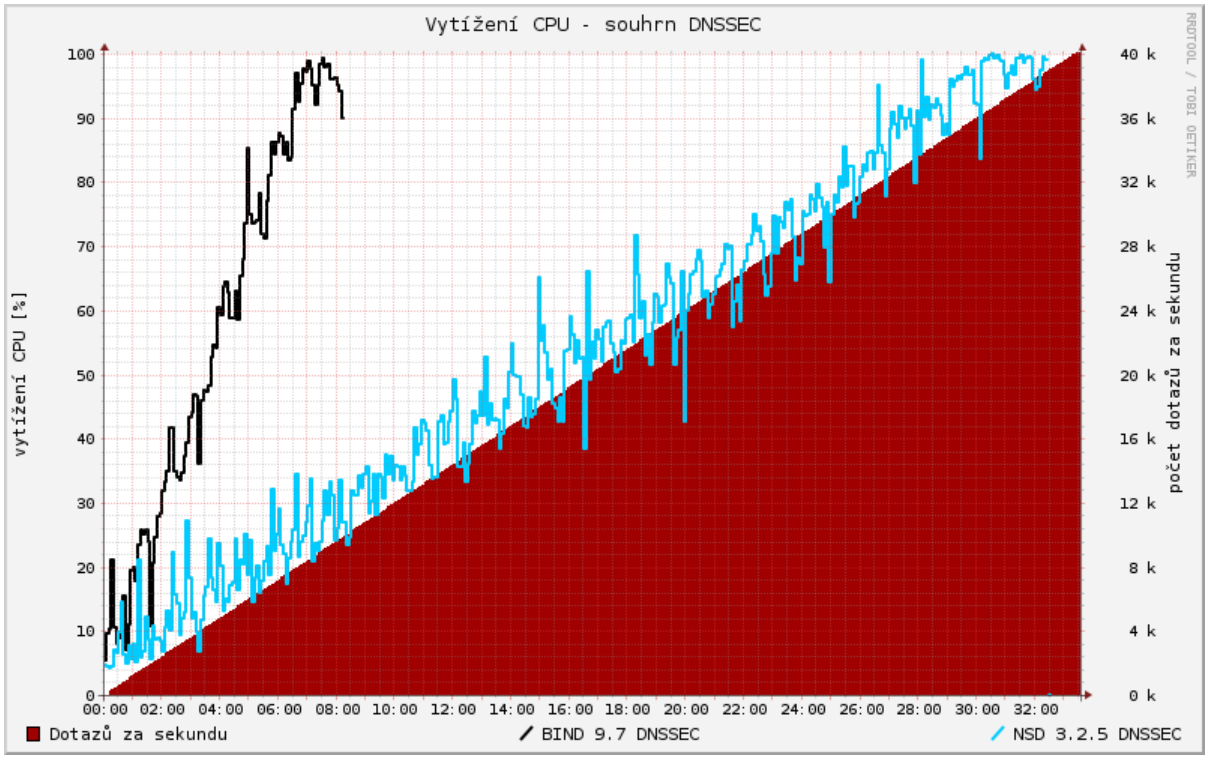
Ilustrace 4.8: Využití CPU – NSD 3.2.22 DNSSEC



Ilustrace 4.9: Využití CPU – PowerDNS 2.9.22



Ilustrace 4.10: Využití CPU – souhrn



Ilustrace 4.11: Využití CPU – souhrn DNSSEC

Výsledky pro implementaci NSD bez technologie DNSSEC (ilustrace 4.7) a s technologií DNSSEC (ilustrace 4.8) jsou si velmi podobné. Graf implementace NSD s technologií DNSSEC je pouze více rozkmitaný kolem pomyslné přímky regrese. Dalším rozdílem je, že implementace NSD se zapnutou technologií DNSSEC trávila více času v systémovém módu a na konci není CPU plně vytíženo. Za tyto rozdíly může určitě zapnutá technologie DNSSEC a rozdílný parametr „maximum number of outstanding queries“. Z grafů lze také vyčíst, že implementace NSD má nejnižší počáteční využití CPU. Jediné co nevím, jak si vysvětlit, je rozkmitanost hodnoty celkového využití CPU u této implementace.

Ilustrace 4.9, zobrazující naměřené výsledky pro implementaci PowerDNS, je nejideálnější ze všech. Výsledky neobsahují žádné výkyvy, vše roste pěkně lineárně. Zajímavostí oproti ostatním výsledkům je poměrně konstantní čas CPU strávený v systémovém módu..

Na ilustracích 4.10 a 4.11 jsou zobrazeny souhrnné grafy, ze kterých se jednodušeji porovnávají jednotlivé implementace mezi sebou. Ve výše zmíněných ilustracích je také navíc vložena druhá osa Y, na níž je vynesen počet dotazů za sekundu pro daný čas. Vytížení CPU na těchto grafech je součtem všech naměřených CPU módů pro danou implementaci, kromě módu idle. Díky tomuto sečtení je ještě patrnější rozkmitanost výsledků u implementace NSD, důvod rozkmitání je mi však neznámý.

Ze zde prezentovaných grafů je jasné, jak si na tom jednotlivé implementace stojí po stránce vytížení CPU při určitém počtu dotazů za sekundu. Z naměřených výsledků vyplývá že, nejefektivněji s CPU nakládá implementace NSD.

4.7.3 Nároky na operační paměť

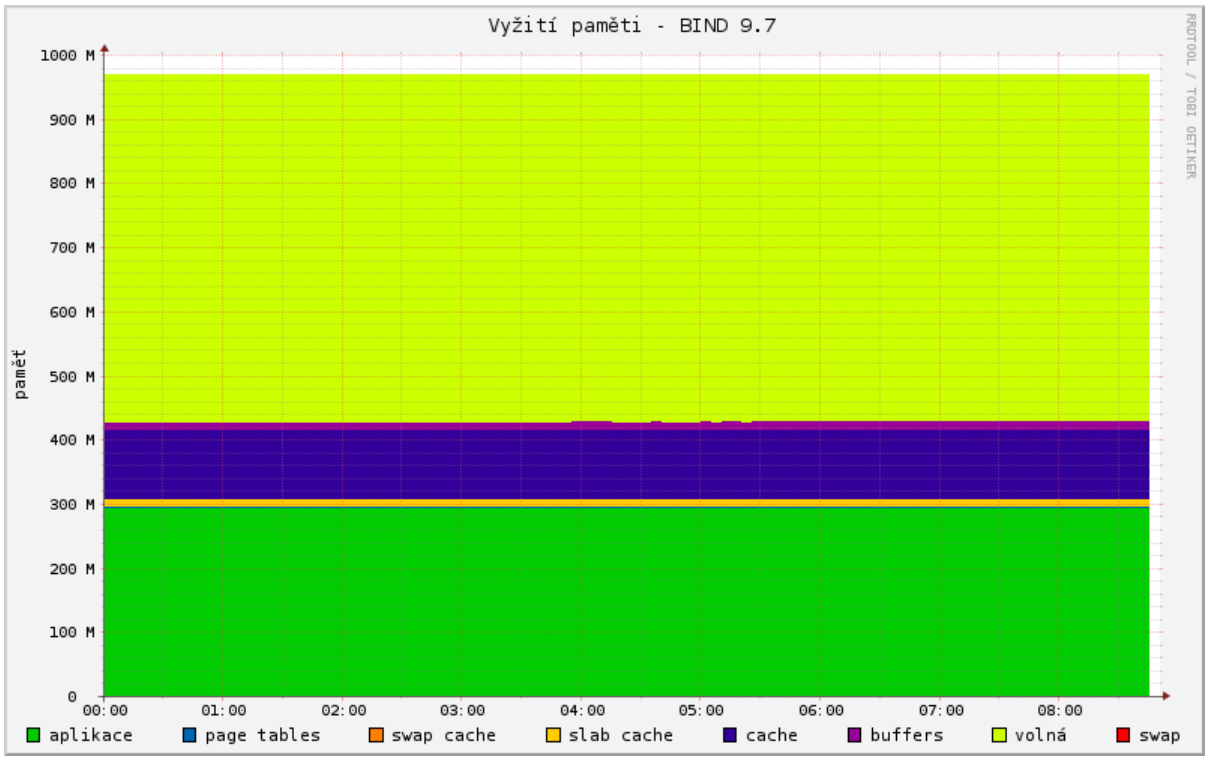
Dalším velmi důležitým ukazatelem je využití operační paměti, protože velcí správci autoritativních DNS serverů mají velké množství rozsáhlých zón a ty samozřejmě spotřebují velké množství operační paměti.

Jelikož jsem měřením zjistil, že využití paměti je až na jednu výjimku u všech implementací konstantní, tak jsem se rozhodl zde uvést pouze jeden detailní a jeden souhrnný graf.

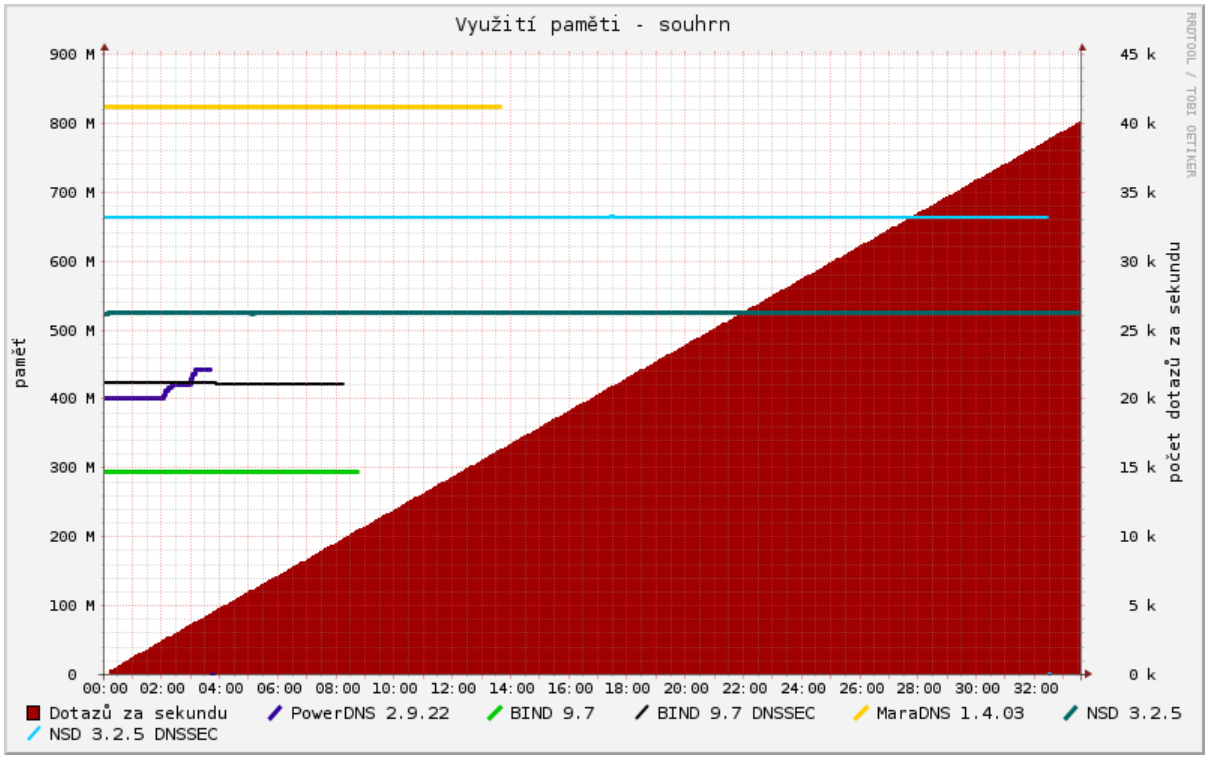
Detailní graf na ilustraci 4.12 je zde jen pro ukázku, abych ukázal jak vypadá výsledek měření paměti. Graf zobrazuje kolik paměti je využíváno aplikacemi, buffery či systémem. Navíc je zde možnost zobrazit velikost odkládací paměti swap, té však nebylo při testech využito.

Souhrnný graf zobrazený na ilustraci 4.13 zobrazuje velikost využité paměti aplikacemi během testování dané implementace. Je na něm vidět, že velikost zabrané paměti se mění pouze u implementace PowerDNS. U PowerDNS si vysvětluji tento nárůst tím, že si interně vytváří vyrovnávací paměti, které jsou přínosné při použití jiných backendů. Další zajímavou věcí, kterou z grafu lze vyčíst, je velikost paměti, kterou jednotlivé implementace potřebují.

V tomto testu dopadla nejlépe implementace BIND, která využila nejméně paměti. Nejhůře dopadla implementace MaraDNS, která využila nejvíce paměti, a to ještě na rozdíl od ostatních spravovala méně záznamů (zhruba o 15 procent).



Ilustrace 4.12: Využití operační paměti – BIND 9.7



Ilustrace 4.13: Využití operační paměti – souhrn

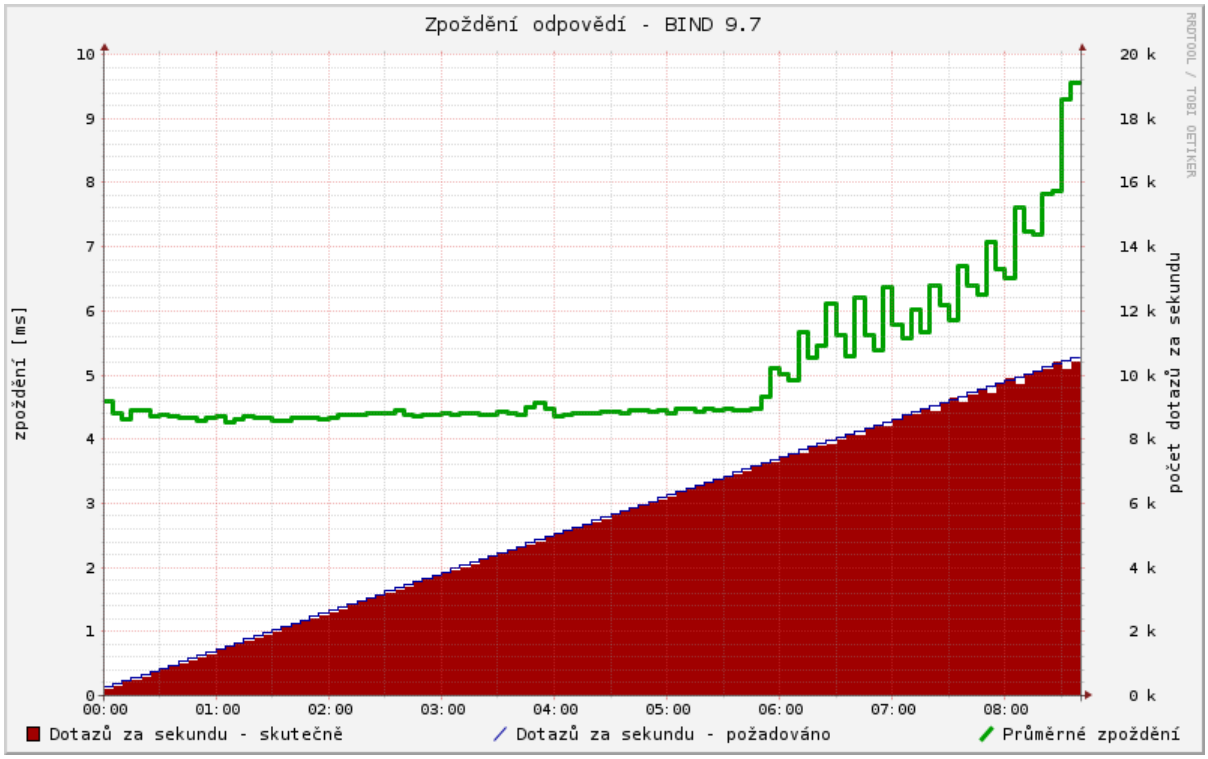
4.7.4 Zpoždění odpovědí (Latence)

Na následujících grafech je zachycena závislost zpoždění odpovědí na počtu dotazů za sekundu. Ještě než se podíváte na grafy, rád bych vysvětlil, proč jsou v grafu dvě hodnoty zobrazující počet dotazů za sekundu. Je to z důvodu parametru „maximum number of outstanding queries“, o kterém jsem se již zmiňoval. Tento parametr zajišťuje „přátelské“ testování implementací. V podstatě se jedná o to, pokud implementace nestíhá obsluhovat dotazy, tak zatěžovací počítač ji nebude přetěžovat, ale „přizpůsobí“ se jejímu tempu.

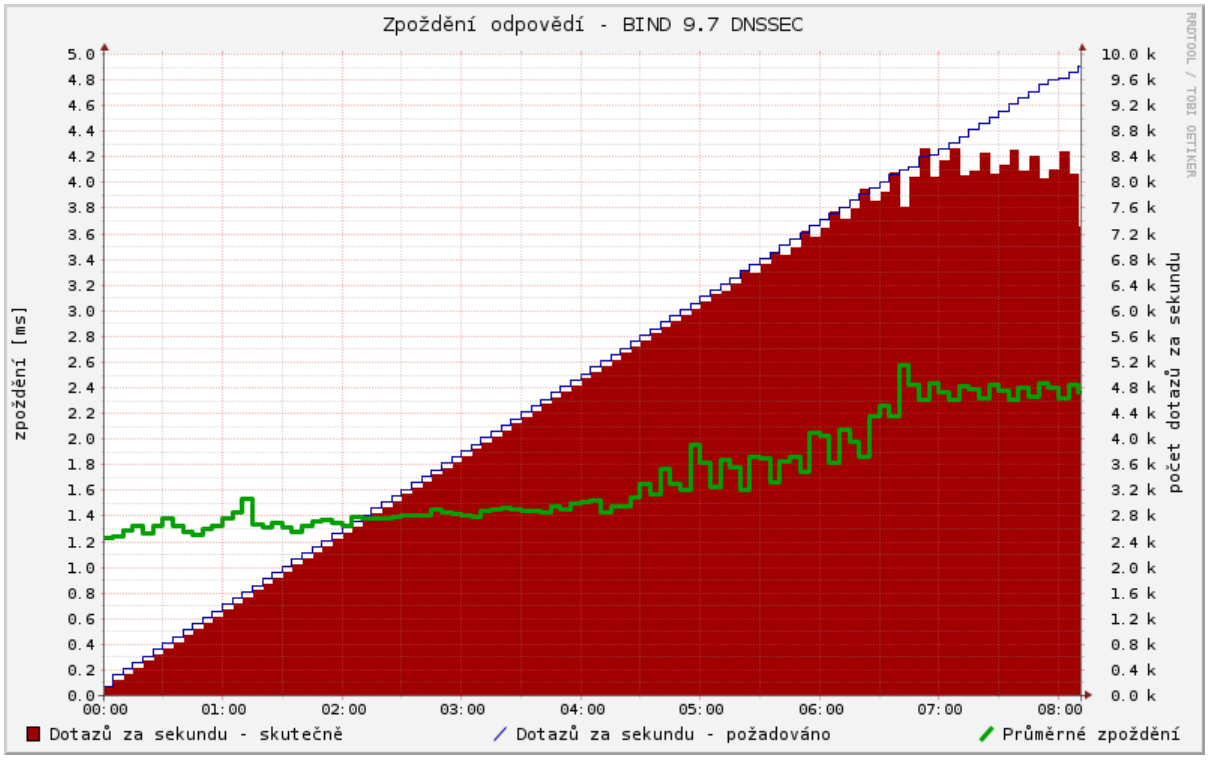
Skutečný počet dotazů říká, kolik opravdu byla za sekundu posláno (a zodpovězeno) dotazů. Zatímco požadovaný počet nám říká, kolik jsme v daný čas chtěli zaslat dotazů. Interpretace pak probíhá následovně. Pokud bylo skutečně posláno méně, než bylo požadováno, pak implementace v tu chvíli ztrácela dech. Pokud byla skutečná hodnota větší než požadovaná, pak se jedná o drobnou chybu měření (špatnou statistiku od měřícího programu). Pokud jsou si obě hodnoty rovny, pak je vše v pořádku.

Zpoždění u implementace BIND (ilustrace 4.14) bylo až do zátěže 7200 dotazů za sekundu zhruba konstantní a pak začalo narůstat mírně exponenciálním tempem, přičemž na konci bylo až dvojnásobné oproti začátku.

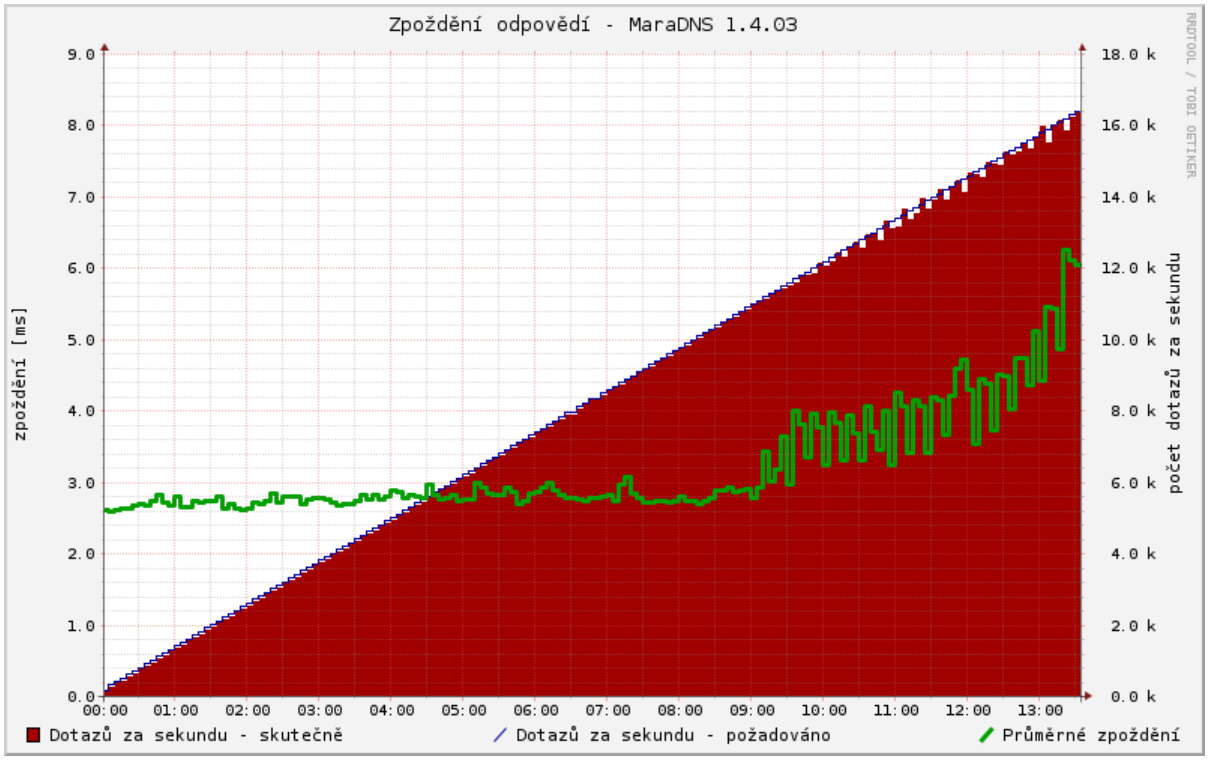
U výsledků implementace BIND se zapnutou technologií DNSSEC (ilustrace 4.15) začalo docházet k nárůstu zpoždění o něco dříve a nárůst byl opět na dvojnásobek. To, že se zpoždění ustálilo, když už se skutečný počet dotazů za sekundu nezvyšoval, naznačuje, že test byl proveden dobře. Je také důležité všimnout si, že zpoždění je oproti stejné implementaci bez zapnutí technologie DNSSEC čtvrtinové. Toto celkové snížení zpoždění je však způsobeno jinou hodnotou parametru „maximum number of outstanding queries“.



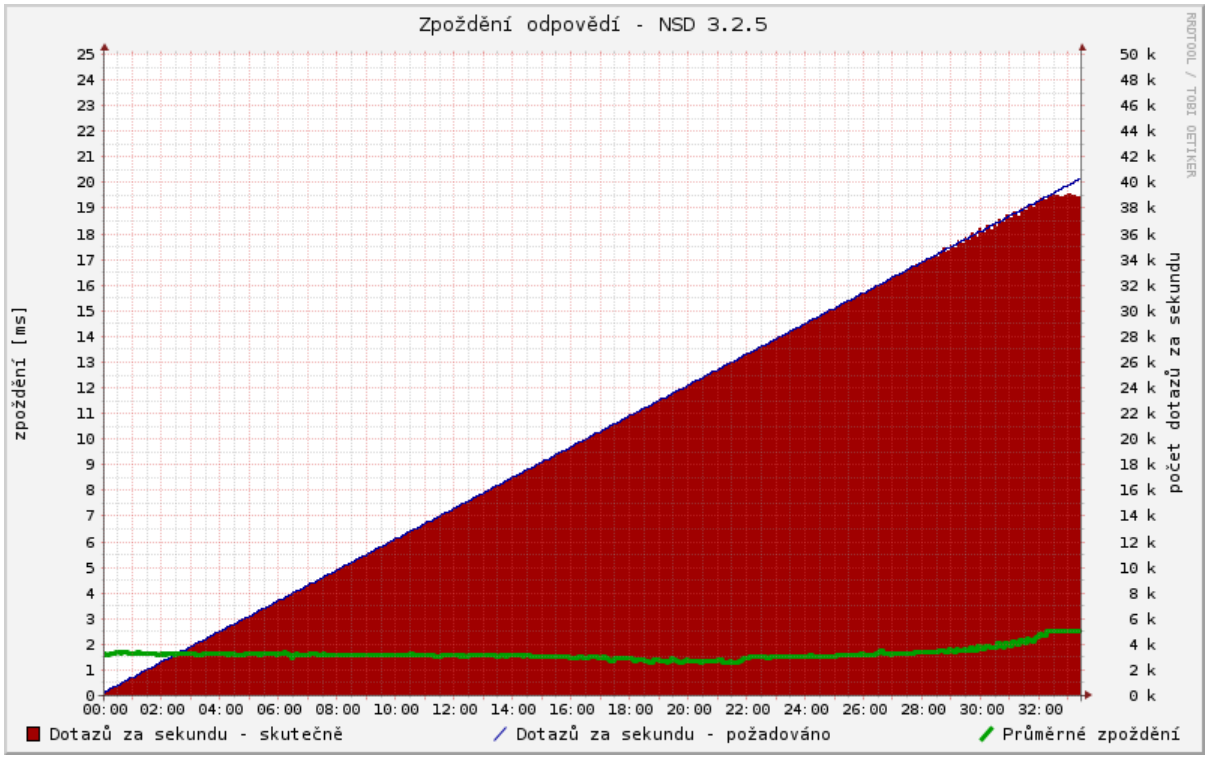
Ilustrace 4.14: Zpoždění odpovědí – BIND 9.7



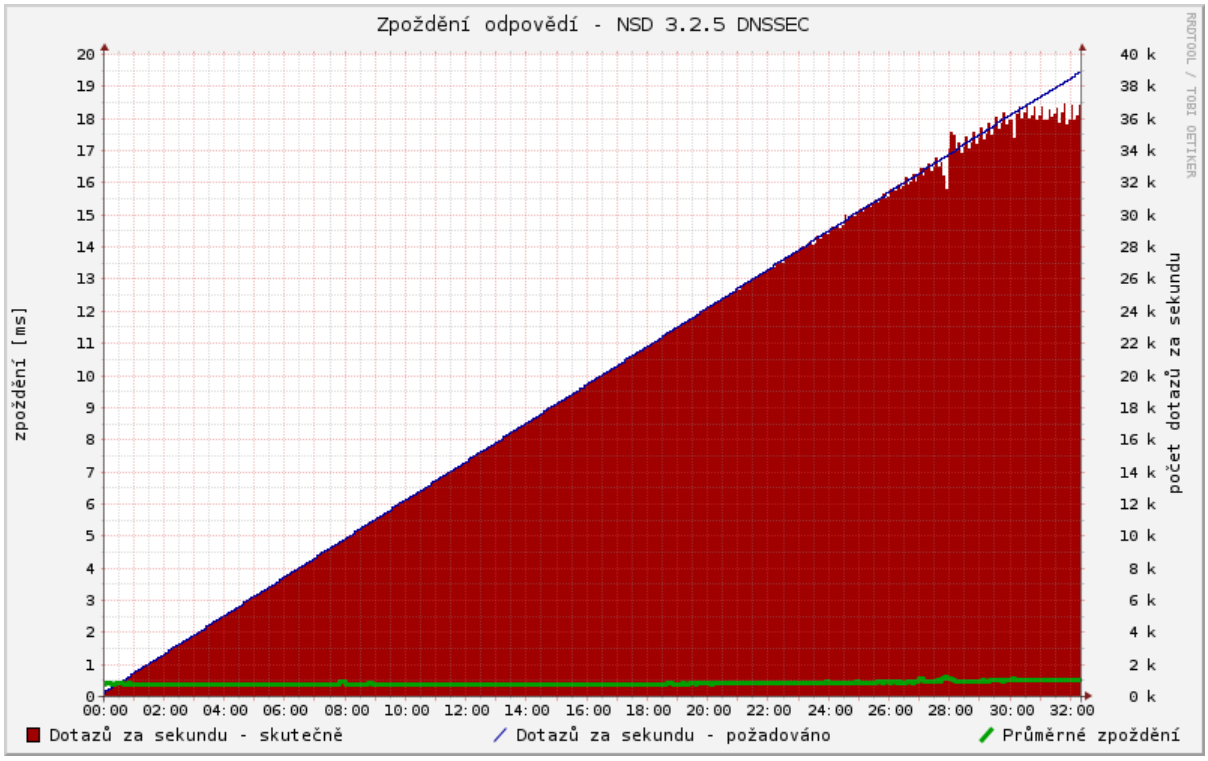
Ilustrace 4.15: Zpoždění odpovědí – BIND 9.7 DNSSEC



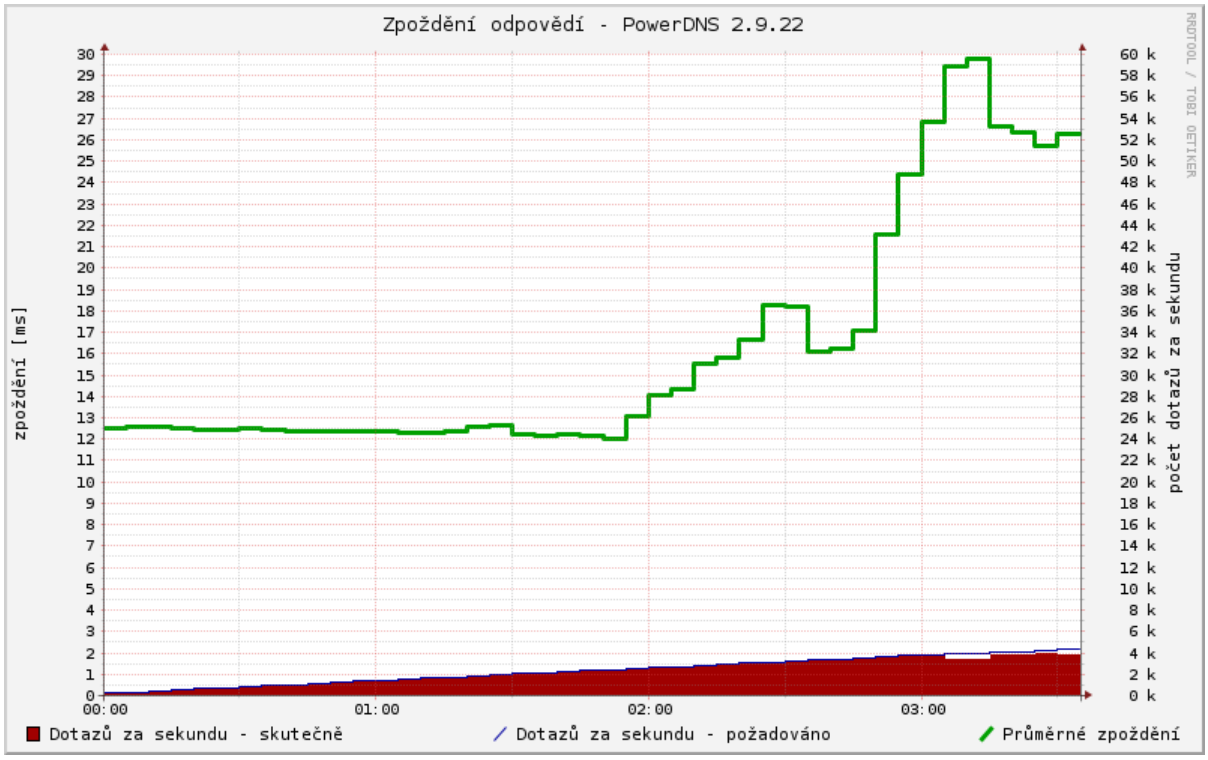
Ilustrace 4.16: Zpoždění odpovědí – MaraDNS 1.4.03



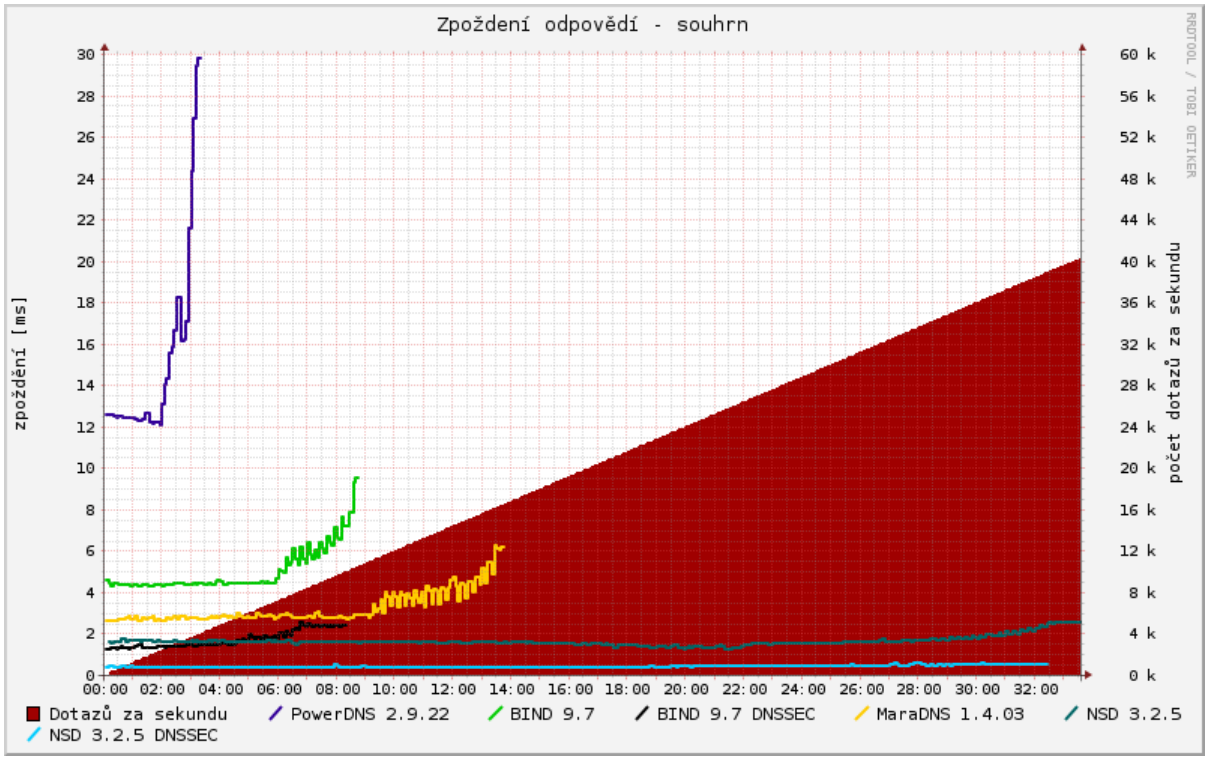
Ilustrace 4.17: Zpoždění odpovědí – NSD 3.2.5



Ilustrace 4.18: Zpoždění odpovědí – NSD 3.2.5 DNSSEC



Ilustrace 4.19: Zpoždění odpovědí – PowerDNS 2.9.22



Ilustrace 4.20: Zpoždění odpovědí – souhrn

Z grafů pro implementace MaraDNS (ilustrace 4.16) a PowerDNS (ilustrace 4.19) je opět vidět, že zpoždění bylo poměrně konstantní do určitého počtu dotazů za sekundu a pak začalo růst. Zejména u implementace PowerDNS byl nárůst 2,5 násobný.

U implementace NSD bez technologie DNSSEC (ilustrace 4.17) byl nárůst zpoždění oproti ostatním implementacím zanedbatelný. Při testování té samé implementace se zapnutou technologií DNSSEC (ilustrace 4.18) k nárůstu zpoždění vůbec nedošlo, opět bych to ale přičetl parametru „maximum number of outstanding queries“. Z výsledků tohoto měření je zajímavé, jak veliký vliv má parametr „maximum number of outstanding queries“ na zpoždění odpovědí.

Jak plyne ze souhrnného grafu na ilustraci 4.20, v tomto testu byla opět nejlepší implementace NSD s minimálním zpožděním a jeho relativní konstantností oproti ostatním implementacím. Ostatní implementace, kromě PowerDNS, mají zpoždění v řádu milisekund a dá se říci, že takto malá zpoždění se v internetu ztratí, jelikož po cestě naberou data větší zpoždění. Zpoždění odpovědí u implementace PowerDNS je dosti velké a nechci si ani představovat, jak velké by bylo, kdyby byl použit jiný backend, například LDAP.

4.7.5 Počet procesů a vláken

Ukazatel počet procesů a vláken se ukázal být při současných testech nic neříkající, protože jeho hodnota byla po celou dobu testu téměř neměnná a nedalo se z naměřených hodnot nic vyčíst. V případě zájmu jsou tyto grafy uloženy na přiloženém optickém médiu.

4.7.6 Disk I/O

Ukazatel vstupních a výstupních operací disku se ukázal být také nezajímavý, jelikož pro všechny implementace měl stejný průběh. Ukazatel ukazoval poměrně periodické zapisování malého množství dat na disk, což si myslím, že bylo způsobené nějakými vnitřními procesy linuxového jádra. V případě zájmu jsou tyto grafy uloženy na přiloženém optickém médiu.

4.7.7 Vyhodnocení

Doufám, že grafy jsou srozumitelné a každý je schopný si z nich vybrat svého vítěze dle svých požadavků (nároků). Obecně se však dá doporučit implementace NSD, která vychází ze všech implementací jako nejlepší. Dále se dá doporučit implementace BIND, která je bezkonkurenčně nejšetrnější k operační paměti a může být vítězem tam, kde je velikost využité operační paměti důležitá.

4.8 Vyhodnocení

Na závěr bych chtěl shrnout všechny testy implementací, proto jsem si připravil tabulku 4.9. Hodnocení je provedeno pomocí znaků, kde dva plusy znamenají velmi dobrý, jeden plus znamená dobrý, lomítko značí průměr a mínus znamená špatný. Výsledné hodnocení je pouze orientační a pro někoho může být nejlepší jiná implementace, kvůli jiným váhám jednotlivých testů. Já bral váhu všech testů jako stejnou. První test pojmenovaný budoucnost, představuje mé subjektivní hodnocení

na základě kapitoly 4.4 Analýza předpokladů a skládá se z použité licence, vývojářské komunity, vývojové historie, historie bezpečnostních chyb, implementované funkčnosti a plánů do budoucna.

	Bind 9.7	MaraDNS 1.4.03	NSD 3.2.5	PowerDNS 2.9.22
Budoucnost	++	-	++	/
Doba startu	+	/	++	-
Maximální zátěž	/	+	++	-
Paměťové nároky	++	-	/	+
Zpoždění odpovědí	/	+	++	-
Výsledné hodnocení	+	/	++	-

Tabulka 4.9: Závěrečné hodnocení implementací v jednotlivých testech benchmarku.

Do budoucna mají nejlepší výhled implementace BIND a NSD, stojí za nimi rozsáhlá komunita, jsou pod svobodnou licencí, počet jejich nasazení je vysoký a jsou to jediné implementace nasazené na kořenových DNS serverech. Nejhůře je na tom implementace MaraDNS, která je vyvíjena pouze jediným člověkem a v současné době postrádá podporu technologie DNSSEC. Implementace PowerDNS, která byla dříve atraktivní hlavně kvůli velkému množství backendů, již ztratila tuto atraktivitu, jelikož i ostatní implementace přidaly podporu backendů.

Nejrychleji startuje implementace NSD, jejíž start je oproti té nejpomalejší více než třikrát rychlejší. Implementací s nejpomalejším startem je PowerDSN.

Nejvíce dotazů za sekundu zvládla obsloužit implementace NSD, její prvenství bylo výrazné, protože druhá nejrychlejší implementace byla o více než polovinu pomalejší. Velmi špatné výsledky měla nejpomalejší implementace PowerDNS, jež byla desetkrát pomalejší než nejrychlejší implementace NSD.

V testech paměti si nejlépe vedla implementace BIND, která zabrala nejméně paměti. Na opačném konci se umístila implementace MaraDNS, která zabrala dvakrát více operační paměti než implementace BIND, a to spravovala méně záznamů než ostatní implementace (viz 4.5.3 Nastavení implementací DNS serverů pro testování).

Nejmenší zpoždění odpovědí jsem naměřil u implementace NSD, u této implementace bylo zpoždění odpovědí poměrně neměnné i při rostoucím počtu dotazů za sekundu. U implementací BIND a MaraDNS bylo naměřeno podobné zpoždění i jeho závislost na počtu dotazů za sekundu, naměřené zpoždění bylo stále poměrně nízké v řádu milisekund. Nejhůře dopadla implementace PowerDNS, u této implementace bylo naměřeno největší zpoždění, zpoždění bylo poměrně konstantní do určitého počtu dotazů za sekundu a pak začalo prudce stoupat až do řádu desítek milisekund.

Celkovým vítězem benchmarku je implementace NSD, která se dle mne bude do budoucna stávat stále populárnější. Implementace BIND je také velice dobrá a vyniká hlavně v nejmenších paměťových nárocích. Bohužel implementaci MaraDNS a PowerDNS nemohu doporučit do žádného velkého prostředí, kde je kladen důraz na výkon, stabilitu a bezpečnost.

5 Závěr

Když jsem poprvé viděl název a požadovaný obsah této bakalářské práce, představoval jsem si ji jako jednoduchou a rutinní záležitost. Velmi brzo se však ukázalo, že práce nebude tak jednoduchá a už vůbec ne rutinní. Tato práce si nakonec vyžádala mnohem více času, úsilí a bezesných nocí než jsem si myslel.

Nakonec jsem s výsledky práce spokojen a mám z ní dobrý pocit. Správnost mých myšlenek a přínosnost výsledků mi byla potvrzena na semináři DNS-OARC konaném v Praze, kde jsem prezentoval výsledky své bakalářské práce a setkal se s převážně kladným ohlasem.

V první části bakalářské práce jsem analyzoval současné metodiky benchmarking autoritativních implementací DNS a navrhl novou komplexní metodiku. Tato nově navržená metodika má mnoho ukazatelů (hodnot k měření) a variant měření. Výsledky metodiky jsou hodnoty a grafické charakteristiky závislosti ukazatelů na daných proměnných.

V druhé části jsem si vybral čtyři implementace autoritativních DNS serverů BIND, NSD, MaraDNS a PowerDNS. Dále jsem prakticky implementoval metodiku definovanou v první části. Následně jsem provedl aplikování (benchmarking) implementované metodiky nad vybranými autoritativními implementacemi DNS. Naměřené výsledky jsem prezentoval formou tabulek či grafů a vyhodnotil. Výsledky měření jsou zajímavé, avšak nikterak šokující. Ukázalo se, že BIND má velmi vážnou konkurenci v NSD, a obě implementace jsou výrazně lepší než zbylé dvě testované. Během testování této metodiky jsem neobjevil žádné problémy, které by ukazovaly, že metodika byla špatně vytvořena. Pouze ukazatel počtu procesů a vláken se ukázal být zbytečný. Samotné testy proběhly bez problémů.

Naměřené výsledky jsou užitečné zejména pro správce větších autoritativních serverů, ti mohou změnou implementace autoritativních DNS serverů ušetřit za nákup nového výkonnějšího HW. Výsledky jim také pomohou lépe poznat jejich implementaci a její chování. Výsledky jsou zajímavější díky tomu, že se mi nepovedlo najít jiné podobně komplexní.

Jako vhodné pokračování této bakalářské práce vidím vytvoření maximálně automatizovaného řešení, které by samo konfigurovalo, spouštělo, monitorovalo, vyhodnocovalo a archivovalo testy. Dále by toto řešení provádělo jejich grafickou a numerickou prezentaci.

Literatura

[1] BORTZMEYER, Stephane. *Generic NIC - DNS registry* [online]. 2004-01-27 [cit. 2010-05-01]. The choices for a nameserver. Dostupné z WWW: <http://www.generic-nic.net/sheets/practical/nameserver-en>.
[2] KNOWLES, Brad. *Shub-Internet* [online]. 2003 [cit. 2010-05-12]. Domain Name Server Comparison. Dostupné z WWW: <http://www.shub-internet.org/brad/papers/dnscomparison/>.
[3] The Measurement Factory. *The Measurement Factory* [online]. 2009-12-30 [cit. 2010-05-01]. DNS SURVEY: OCTOBER 2009. Dostupné z WWW: <http://dns.measurement-factory.com/surveys/200910.html>.
[4] MOORE, Don. *MyDNS* [online]. 2004-05-23 [cit. 2010-05-01]. DNS server survey. Dostupné z WWW: <http://mydns.bboy.net/survey/>.
[5] TRENHOLME, Sam. New revised MaraDNS roadmap. *Musings about MaraDNS and technology* [online]. 2010-05-01, [cit. 2010-05-01]. Dostupný z WWW: <http://maradns.blogspot.com/2010/04/new-revised-maradns-roadmap.html>.
[6] KABELOVÁ, Alena; DOSTÁLEK, Libor. *Velký průvodce protokoly TCP/IP a systémem DNS* . Vyd. 5. Praha : Computer Press, 2008. 488 s. ISBN 978-80-251-2236-5.

Seznam příloh

Příloha 1. DVD

Přiložené DVD obsahuje:

- měřící programy, skripty a nástroje,
- naměřená data ve formátu RRD,
- konfigurační soubory pro implementace,
- vygenerované zóny DNS,
- vygenerované konfigurační soubory s dotazy DNS.